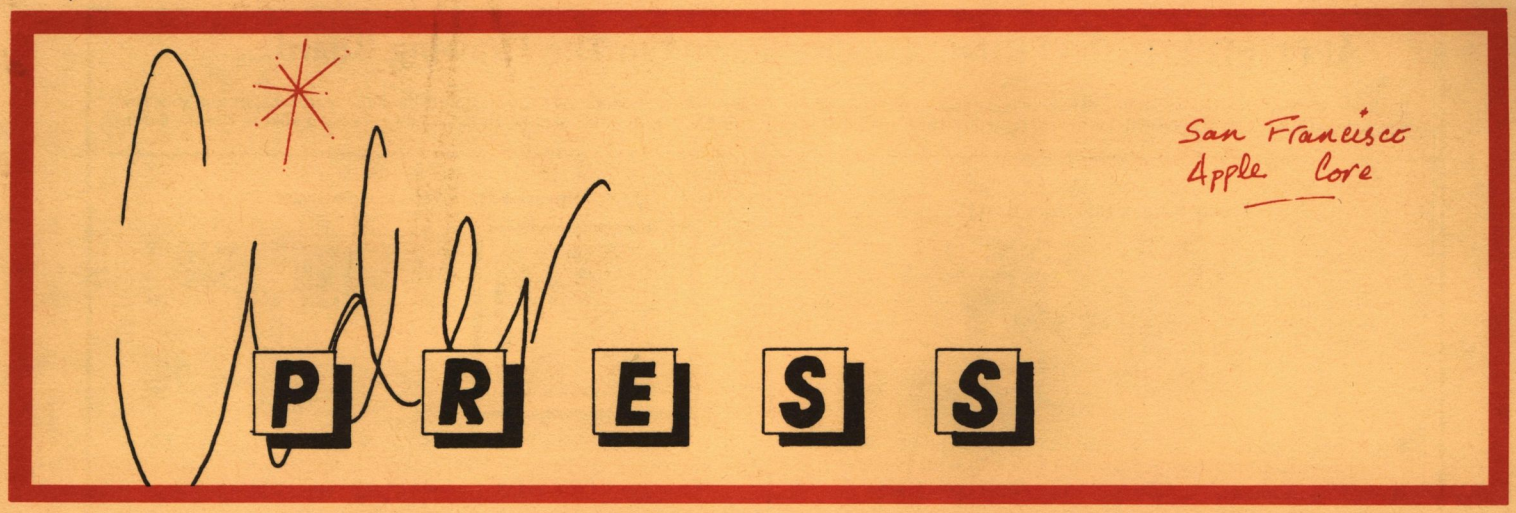


San Francisco Apple Core

IN THIS ISSUE

B.A.S.I.C. - Joyce Conklin 2
F.M.L.I. - Bryan Costales 9
Machinations - Don Scellato 12
Brass Apple Award - Don Scellato 21
ReWrite - Preliminary Review - Allan Woods 24
Programming Contest - Don Scellato 25
Nu'ers - Kathy Rudovsky 26
Macintosh Interest Group - Don Scellato 27
Interest Group News - Kathy Rudovsky 27
Calendar 28
Interest Group Leaders 29
VisiCalc Interest Groups - Don Scellato 30
Poke of the Month - Kathy Rudovsky 30
Meeting Map 31
Membership Application 32

GENERAL MEETINGS:

June 23

July 20

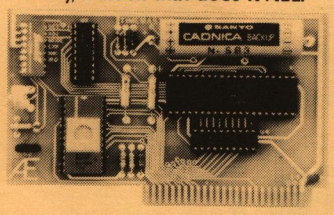
May/June, 1984

APPLIED ENGINEERING Quality Starts On The Drawing Board....

But it doesn't end there. Our commitment to excellence applies all the way down the line, through manufacturing and testing to our three year no-hassle warranty. The aim, of course, is to provide you with the best designed and best crafted hardware available. Try any of the peripherals below and see what it's like to own the finest.

The TIMEMASTER
Finally, a clock that does it ALL!

Automatically date stamps files with PRO-DOS



Fully PRO-DOS compatible

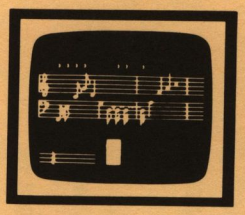
- Just plug it in and your programs can read the year, month, date, day, and time to 1 millisecond! The only clock with both year and ms.
- A rechargeable NiCad battery will keep the TIMEMASTER running for over ten years.
- Powerful 2K ROM driver — No clock could be easier to use.
- Full emulation of most other clocks, including Mountain Hardware's Appleclock (but you'll like the TIMEMASTER mode better).
- Basic, Machine Code, CP/M and Pascal software on 2 disks!
- Eight software controlled interrupts so you can execute two programs at the same time (many examples are included).
- On-board timer lets you time any interval up to 48 days long down to the nearest millisecond.

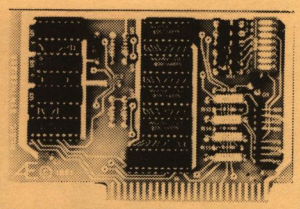
The TIMEMASTER includes 2 disks with some really fantastic time oriented programs (over 25) plus DOS dater so it will automatically add the date when disk files are created or modified. The disk is over a $200.00 value alone — we give the software others sell. All software packages for business, data base management and communications are made to read the TIMEMASTER.

If you want the most powerful and the easiest to use clock for your Apple, you want a TIMEMASTER.

PRICE $129.00

Super Music Synthesizer
Improved Hardware and Software

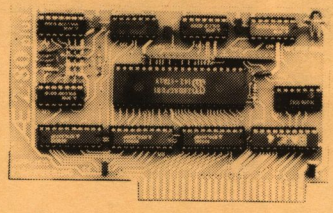




- Complete 16 voice music synthesizer on one card. Just plug it into your Apple, connect the audio cable (supplied) to your stereo, boot the disk supplied and you are ready to input and play songs.
- It's easy to program music with our compose software. You will start right away at inputting your favorite songs. The Hi-Res screen shows what you have entered in standard sheet music format.
- Now with new improved software for the easiest and the fastest music input system available anywhere.
- We give you lots of software. In addition to Compose and Play programs, 2 disks are filled with over 30 songs ready to play.
- Easy to program in Basic to generate complex sound effects. Now your games can have explosions, phaser zaps, train whistles, death cries. You name it, this card can do it.
- Four white noise generators which are great for sound effects.
- Plays music in true stereo as well as true discrete quadraphonic.
- Full control of attack, volume, decay, sustain and release.
- Will play songs written for ALF synthesizer (ALF software will not take advantage of all our card's features. Their software sounds the same in our synthesizer.)
- Our card will play notes from 30HZ to beyond human hearing.
- Automatic shutoff on power-up or if reset is pushed.
- Many many more features.

PRICE $159.00

Z-80 PLUS!



- TOTALLY compatible with ALL CP/M software.
- The only Z-80 card with a special 2K "CP/M detector" chip.
- Fully compatible with microsoft disks (no pre-boot required).
- Specifically designed for high speed operation in the Apple IIe (runs just as fast in the II+ and Franklin).
- Red "CP/M WORKING" LED indicator, the Z-80 Plus does not interfere with non-CP/M programs.
- A semi-custom I.C. and a low parts count allows the Z-80 Plus to fly thru CP/M programs at a very low power level. (We use the Z-80A at fast 4MHZ.)
- Does EVERYTHING the other Z-80 boards do, plus Z-80 interrupts.

Don't confuse the Z-80 Plus with crude copies of the microsoft card. The Z-80 Plus employs a much more sophisticated and reliable design. With the Z-80 Plus you can access the largest body of software in existence. Two computers in one and the advantages of both, all at an unbelievably low price.

PRICE $139.00

Viewmaster 80

There used to be about a dozen 80 column cards for the Apple, now there's only ONE.

- TOTALLY Videx Compatible.
- 80 characters by 24 lines, with a sharp 7x9 dot matrix.
- On-board 40/80 soft video switch with manual 40 column override
- Fully compatible with ALL Apple languages and software—there are NO exceptions.
- Low power consumption through the use of CMOS devices.
- All connections on the card are made with standard video connectors.
- All new design (using a new Microprocessor based C.R.T. controller) for a beautiful razor sharp display.
- The VIEWMASTER incorporates all the features of all other 80 column cards, plus many new improvements.

	PRICE	BUILT IN SOFTSWITCH	SHIFT KEY SUPPORT	LOW POWER DESIGN	THREE YEAR WARRANTY	7x9 DOT MATRIX	LIGHT PEN INPUTS	40 COLUMN OVERRIDE	INVERSE CHARACTERS
VIEWMASTER	**169**	**YES**	**YES**	**YES**	**YES**	**YES**	**YES**	**YES**	**YES**
SUP'RTERM	MORE	NO	YES	NO	NO	NO	NO	YES	YES
WIZARD80	MORE	NO	NO	NO	NO	YES	NO	YES	YES
VISION80	MORE	YES	YES	NO	NO	YES	NO	NO	NO
OMNIVISION	MORE	NO	YES	NO	NO	NO	NO	YES	YES
VIEWMAX80	MORE	YES	YES	NO	NO	YES	NO	NO	YES
SMARTERM	MORE	YES	YES	NO	NO	NO	NO	YES	YES
VIDEOTERM	MORE	NO	NO	YES	NO	YES	YES	NO	YES

The VIEWMASTER 80 works with all 80 column applications including CP/M, Pascal, WordStar, Format II, Easywriter, Apple Writer II, VisiCalc, and all others. The VIEWMASTER 80 is THE MOST compatible 80 column card you can buy at ANY price!

PRICE $169.00

MemoryMaster IIe 128K RAM Card

- Expands your Apple IIe to 192K memory.
- Provides an 80 column text display.
- Compatible with all Apple IIe 80 column and extended 80 column card software (same physical size as Apple's 64K card).
- Can be used as a solid state disk drive to make your programs run up to 20 times FASTER (the 64K configuration will act as half a drive).
- Permits your IIe to use the new double high resolution graphics.
- Automatically expands Visicalc to 95 K storage in 80 columns! The 64K config. is all that's needed, 128K can take you even higher.
- PRO-DOS will use the MemoryMaster IIe as a high speed disk drive.
- Lowest power consuming 128K card AVAILABLE.
- Complete documentation included, we show you how to use all 128K.

If you already have Apple's 64K card, just order the MEMORYMASTER IIe with 64K and use the 64K from your old board to give you a full 128K. (The board is fully socketed so you simply plug in more chips.)

MemoryMaster IIe with 128K	**$249**
Upgradeable MemoryMaster IIe with 64K	**$169**
Non-Upgradeable MemoryMaster IIe with 64K	**$149**

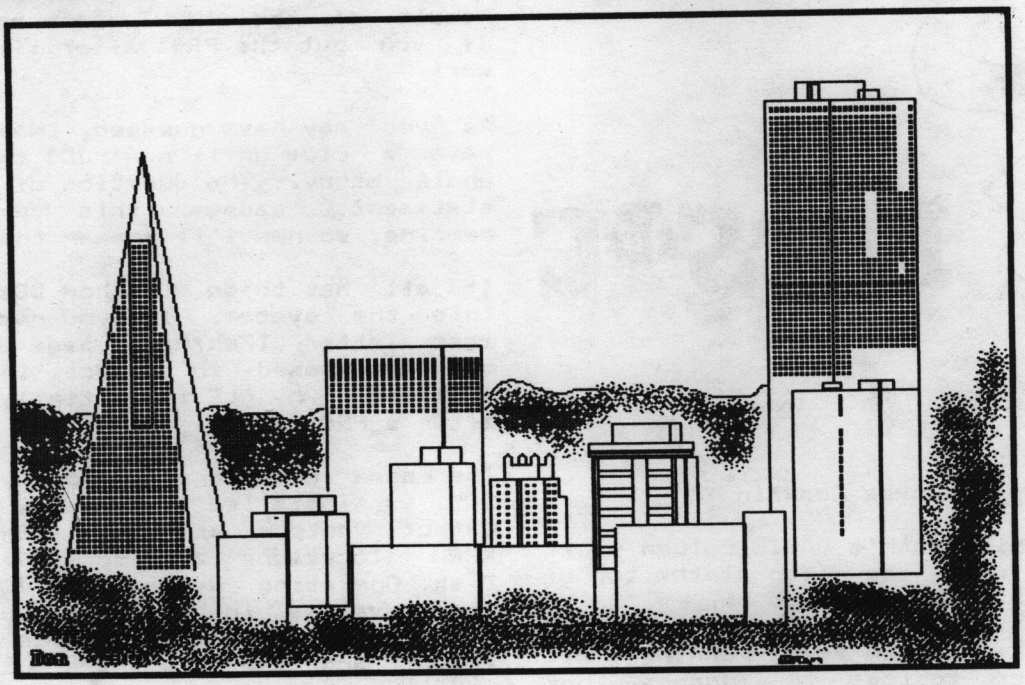
Our boards are far superior to most of the consumer electronics made today. All I.C.'s are in high quality sockets with mil-spec. components used throughout. P.C. boards are glass-epoxy with gold contacts. Made in America to be the best in the world. All products work in the APPLE IIE, II, II+ and Franklin. The MemoryMaster IIe is IIe only. Applied Engineering also manufactures a full line of data acquisition and control products for the Apple; A/D converters and digital I/O cards, etc. Please call for more information. All our products are fully tested with complete documentation and available for immediate delivery. All products are guaranteed with a no hassle **THREE YEAR WARRANTY.**

All Orders Shipped Same Day
Texas Residents Add 5% Sales Tax
Add $10.00 If Outside U.S.A.
Dealer Inquiries Welcome

Send Check or Money Order to:
APPLIED ENGINEERING
P.O. Box 470301
Dallas, TX 75247

Call (214) 492-2027
7a.m. to 11p.m. 7 days a week
MasterCard, Visa & C.O.D. Welcome
No extra charge for credit cards

CIDER PRESS



THE CORE'S CORE

Directors

Sami Capron	415/388-8631
Joyce Conklin	415/341-6955
Michael Fischer	415/933-4435
Alan Morris	415/566-2342
Kathy Rudovsky	415/756-6480
Don Scellato	415/672-2051
Alan L Smerican	415/566-2342
Allan B. Woods	415/687-9659

Service/Committees

Membership	Alan L Smerican Ted Chatterton	415/566-2342
Cider Press -		
Editor	Allan Woods	415/687-9659
BASIC	Joyce Conklin	415/341-6955
Marketing Dir.	Don Scellato	415/672-2052
General Meeting	Michael Fischer	415/933-4435
Treasurer	Alan Morris	415/566-2342
Interest Groups	Kathy Rudovsky	415/756-6480

CIDER PRESS POLICY

All manuscripts, photographs, and other materials will be considered submitted free and released for publication. They become the property of the San Francisco Apple Core and the CIDER PRESS. Authors should clearly mark all material submitted for publication so that proper credit can be given.

The views expressed in any articles are those of the authors, and not necessarily those of the San Francisco Apple Core, Inc., the Editor, Club Officers or Directors . . . or anyone else in their right minds.

The San Francisco Apple Core Incorporated is a non - profit organization comprised of and supported by owners and users of Apple I I and Apple / / / computers. The Club is run by officers and committees, and the Club endeavors to aid Apple owners through its Program Libraries, Program Disks, meetings, and this publication.

Article Submission: The CIDER PRESS welcomes any and all articles dealing with Apple computers and their associated hardware and software. Articles may be submitted (typed or printed, double - space), or as disk text files using any of the popular word processors or text editors. Any author who has an article published in the CIDER PRESS will receive a free copy of the most current Disk - of - the - Month.

Reprints: All articles appearing in the CIDER PRESS not copyrighted by the author may be reprinted by other *non - profit* Apple User Groups, without prior permission, so long as proper credit is given to both the author and the CIDER PRESS. Permission to reprint a copyrighted article must be obtained in advance, by writing to the authors, c/o the San Francisco Apple Core.

San Francisco Apple Core
1515 Sloat Blvd.
San Francisco, CA 94132

B*A*S*I*C

Copyright, 1984 by Joyce Conklin

Welcome to this month's BASIC column -- at least that's what the logo at the top of the page says it's supposed to be.

After two years, I seem to be running a bit dry on what topics to address, but fortunately I always can make a new mistake which will require asking a question, and some of my good techie (that is not a perjoritive in my vocabulary!) friends will respond with an answer.

Both Stan Dinsmore and Morris Prosser came through with answers to my question in the Feb/Mar issue on page 4, concerning the use of PR#1 to turn on the printer and its placement in a program. It turns out that my problem is that I still don't understand all the ins and outs of DOS - when I first got my II+ system I dutifully sat down to read the manuals and try the examples.

Applesoft Tutorial was fine (except for a couple of typos in the manual which I was able to decipher), and my Applesoft Reference Manual is well thumbed and flagged with paper-clips because I use it all the time (still!).

But . . . when I started to read the DOS manual I got as far as page 83 in the Random Access Files chapter (I can tell how far I read because that is the last page with yellow highligter o it!) and pretty well gave up. Looking through it just now to find that page I see that there is an illustration of track and sector allocation in the DOS manual - if I did see it before I didn't know what I was looking at!

All of this is by way of encouragement to those who want to learn programming or just more about the Apple but think that they'll never make the grade. Have patience - it will become clearer with time.

Explanation by Morris Prosser

The problem you encountered in your random list program was not the result of the HOME statement in line 10. Rather, it was the result of the INPUT statement in line 30. If you put the PR#1 after line 30, it will work.

As you may have guessed, [Nope -- I didn't have a clue until now! JC] that is not the whole story. The question of why the INPUT statement causes this problem still remains, so now I'll answer that.

It all has to do with how DOs is connected into the system. IF you remember, in the last letter [Feb/Mar - page 3, answering a question posed in the Oct. issue, p.6-7] I explained why PLE is partially disconnected after a PR#1.

The same considerations apply to DOS. When DOS is installed it looks at the input and output vectors at CSWL, CSWH, KSWL, and KSWH (locations $36 to $39). It [DOS or Disk Operating System .. JC] then stores those values in DOS memory space (at $AA53 to $AA56) and substitutes its own input and output addresses for the output and input vecotrs.

It then monitors all input and output to the compter to see whether it is required to take any action. Like PLE, if one o these vectors is disconnected and the other is used, the one that was disconnected is reconnected.

Now let's consider the result of all this in the spedific case of your program, and what you can do to prevent problems of this kind. If you put PR#1 in line 5, output is routed to the printer.

When you get an input in line 30, however, the input comes through DOS, which notices that its input (which screens [or analyzes -- JC] program output) has been disconnected. So it reconnects its input and sends its output to the address it has recorded previously (the video screen), thus disconnecting the printer.

If you put the PR#1 in line 95, the program works all right, since there is no input to the program coming through DOS before the printout. You can verify that DOS is partially disconnected at this point by adding a line:

```
120 PRINT CHR$(4) "CATALOG"
```

The way to avoid these problems, (an the

way PR# and IN# should always be used in connection with DOS), is to use a control-D before issuing the command. This alerts DOS to the change, and it resets all the vectors accordingly. One way to do this is to use the lines:

```
5 PRINT CHR$(4) "PR#1"
```

and

```
110 PRINT CHR$(4) "PR#0"
```

This keeps DOS happy and makes the program work the way you meant it to.

On another subject, I noticed that [there were some] typos on ... the "universal algorithm." [#2 on p.3 of the Feb/Mar issue, continuing over to p.4. JC] The first Boolean expression on the second line as published should be "(Q > 0)." Another typo ... in the published version - the Boolean expression in the third line should be "(Q > 7)."

Many thanks Morris, and also Stan. For the benefit of those who are keeping track of the programs printed in the BASIC column, here is line 520 from the "universal algorithm" program, as it should appear:

```
520 Q = INT ((SC(L) / N) * 10 + .05) - 3:
    Q = Q * (Q > 0) * (Q > 7) + (Q < 1) +
    6 * (Q < 6) : FD(Q) = FD(Q) + 1
```

New Topic

I received a call from a new member of the club, Magin (NLN as they say in police parlance - you didn't give me your last name, Magin!) asking if there were a way to disconnect the 80-column card from within AppleWriter //e so that the display would appear in the larger-type 40-column mode.

The reason for wanting to do this was to make it a bit easier on the eyes after many hours at the keyboard and monitor. I, too, have experienced the bleariness of VDT fatigue. Another collegue, Pat Malone, also said that she had been searching for the same solution but had found that the only answer was to pull the 80-column card!

Well, that will certainly get you into the 40-column mode. My suggestion was that both of them use AppleWriter II on the //e; because the II version will not recognize the 80-column card in the auxilliary slot, the display will be in 40-column size, but any files constructed will be usable by AppleWriter //e as well since they are all Apple text files (the kind denoted by a T when you catalog a disk.)

An Interesting HELLO Progam

One of my student aides has designed a HELLO program which will prevent unauthorized users from accessing your disk of wonderful stuff. I will give you the listing and attempt to explain some of the esoterica.

```
10  REM HELLO PROGRAM DONATED BY

15  REM *************************
17  REM *                       *
20  REM *   Richard Goldstein   *
30  REM *                       *
40  REM *************************

50  POKE 1012,0

55  REM What's Where in the Apple by W.F.
    Leubbert says this location (1012) is
    the "Autostart ROM Power up Mask

60  TEXT : HOME

70  POKE 50,128

75  REM Line 70 causes a cursor to appear
    in the upper left corner but not move
    with keyboard input; 128 is the ASCII
    code for the nul or Ctrl-@ character

80  INPUT A$

90  IF A$ = "QWERTY" THEN  NORMAL :
    GOTO 120

95  REM Line 90 checks for proper pass-
    word

100  FOR I = 1 TO 10: READ T: POKE 767
     + I,T: NEXT I: CALL 768

105  REM Lines 100-110 = tone Routine for
     improper password

110  DATA  32,96,251,32,58,255,76,3,3,0

115  REM Lines #120 and on = Routine to
     list catalog entries by letter

120  TEXT : HOME :D$ =  CHR$ (4): PRINT
     D$"CATALOG":B =  PEEK (37) - 2:
     IF B > 22 THEN B = 22

130 T = 0:CH = 4: FOR CV = 0 TO 23:
    GOSUB 260: IF C <  > 160 THEN
    POKE P - 1,219: POKE P,T + 193:
    POKE P + 1,221:T = T + 1:S = CV

140  NEXT CV: VTAB 24:A$ = "TYPE LETTER
     TO RUN, OR LOAD=1 LOCK=2 UNLOCK=3
     DELETE=4 EXIT=5..."

150 B$ = "RUN": HTAB 1: PRINT  LEFT$
    (A$,39);:A$ =  MID$ (A$,2) +  LEFT$
    (A$,1):K =  PEEK ( - 16384): IF K <
    128 THEN  FOR K = 1 TO 75: NEXT K:
    K =  FRE (0): GOTO 150

155 REM location -16384 is the keyboard
    strobe - PEEK there to see what was
    the last key pressed
```

```
160  POKE  - 16368,0:K = K - 176: IF
     K < 1 OR K > 5 THEN 230

165  REM location -16368 clears the key-
     board strobe

170  HTAB 1: CALL  - 868: IF K = 5 THEN
     END
175  REM Call -868 clears to the end of
     the line

180  PRINT " PRESS 'LETTER' YOU WISH
     TO  "; : IF K = 1 THEN B$ = "LOAD"

190  IF K = 2 THEN B$ = "LOCK"

200  IF K = 3 THEN B$ = "UNLOCK"

210  IF K = 4 THEN B$ = "DELETE": FLASH

220  PRINT B$; : CALL  - 198: NORMAL :
     GET K$:K =  ASC (K$) - 48

225  REM Call -198 beeps the speaker

230  IF K < 17 OR K > T + 16 THEN 150

240 CH = 1:CV = S - T + K - 16: GOSUB
    260: IF C = 194 AND (B$ = "RUN" OR
    B$ = "LOAD") THEN B$ = "B" + B$

250  FOR CH = 6 TO 39: GOSUB 260:B$ = B$
     +  CHR$ (C): NEXT CH: HTAB 1:
     CALL  - 868: PRINT B$: PRINT D$;B$:
     GOTO 120

260 C1 =  INT (CV / 8):C2 = CV - C1 * 8:
    P = 1024 + 128 * C2 + 40 * C1 + CH:
    C =  PEEK (P): RETURN
```

An Item from the April Meeting

The "demonstrations" at this meeting were all from club members (actually, Michael Fischer and myself) because the planned demonstrations were unable to come at the last minute.

The first hour was devoted to answering questions and one concerned the use of FID, the program for copying DOS 3.3 files found on your Apple II+ System Master disk. After showing how to start the program and declare the disk drives being used, Michael showed the use of the equals sign (=) as a wildcard so that you don't have to know the file names -- you can just step through the catalog of the source disk and answer Y(es) or N(o) to each.

I spoke up and said that once you had used the = sign, you couldn't get out until you'd answered each of the files on the source disk with Y or N. I'm glad that I wasn't the only one who didn't know about that "undocumented" feature.

Actually, it is documented in the Apple II+ DOS Manual (see above about why I didn't know about it!) on page 185. (This same information is found in the DOS User's Manual supplied with the Apple //e on p. 85-86.)

An Interesting Problem

While reading one of my many computer books in search of more material for the isatiable maw of student learning (well, maybe only "moderately interested" maw!!) I came across the following problem in "Theory and Problems Computers and Programming" by Francis Scheid, one of the Schaum's Outline Series. Work with it over the next month, as I will, and let's see what we come up with.

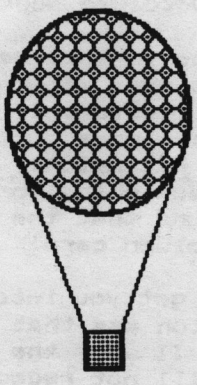
"A coin is tossed until heads appears for the first time. A payoff is then made, equal to the total number of tosses. Probability theory shows that the average amount won over the long run will be...

1/2(1) + 1/4(2) + 1/8(3) + 1/16(4) + ...

where the dots indicate that the computation just goes on and one. Develop a program to sum up the first 20 terms of this sum, printing out the partial results along the way.

Observe the steady shrinkage of the individual terms and the "convergence" of the sums toward the exact average payoff. What does this payoff seem to be?"

This particular book is language independent, so you can see examples in FORTRAN, PL/I and Pascal as well as BASIC.



B.A.S.I.C.

Copyright (c) 1984, by Joyce Conklin

The following program listing is for a rather interesting HELLO program, which was given to me by one of my student aides. While I was trying to save it as a text file (see the BASIC column, Oct. '83 Cider Press) I needed to renumber the original program, which started with line #1 instead of the present line #100. On my working disk I have PLE, which loads upon booting. I also have RENUMBER which I need to load separately.

(Techie question #1 for the month: does anyone know the way to code a HELLO program so that it will RUN two or more programs upon boot-up?)

I ran PLE and RENUMBER so that they were both in memory above HIMEM. Then I loaded the HELLO program from another disk. I had to type three control letters in the program name (the original author hid those letters -- A, S, and D) so that not just anyone could investigate the coding. Well, it seems that the use of Control-D (at least I assume that that is the culprit) "unhooked" my PLE.

(Techie question #2: does this sound like a logical explanation?)

At any rate, I had a great deal of difficulty getting the program to be co-resident with RENUMBER and PLE, but if I only used RENUMBER, there was no problem. There are times when PLE is a lot of extra trouble, but it sure is useful when you are coding your own programs.

The current version of this programmer's aid is called GPLE and is available from Beagle Brothers. But beware ... if you plan to do your programming on the new Apple //c which was demonstrated at the May general meeting, double check the use of GPLE. The copy which I have, purchased from Call A.P.P.L.E. before Beagle Bros. took over the distribution, works on the II Plus and on the //e but NOT on the //c.

Some of the coding in this program is obscure to me, so I would appreciate some help from my techie friends (Morris, are you listening??!!). Location 40286 and 40287 (lines 100,110) are the entry points for soft-entry for Applesoft (according to my copy of "What's Where in the Apple" by Wm. F. Leubbert, published by Micro Ink) but I don't know what the pokes are supposed to accomplish.

```
100  POKE 40286,35

110  POKE 40287,216

120  ONERR  GOTO 310

130  TEXT : HOME

140  HTAB 16: PRINT "APPLE ]";

150  PRINT  CHR$ (91)
```

The above line will allow you to print the left bracket even with an Apple II Plus, which does not have that character on the keyboard.

```
160  VTAB 7: INPUT "PLEASE ENTER
     CODE -. ";AC$

170  SPEED= 100

180  VTAB 7

190  PRINT "
                "
```

There are 40 spaces between the quotes.

```
200  IF AC$ = "JOSHUA" THEN 230

210  IF AC$ = "RG" THEN 270
```

The passwords in the above two lines may be changed to anything you want; obviously Rich Goldstein, the contributer of this program, had seen "Wargames"!

```
220  GOTO 310

230  VTAB 7: PRINT "ONE MOMENT PLEASE...."

240  FOR X = 1 TO 4000: NEXT

250  VTAB 10: PRINT "YOU ARE CLEARED"

260  FOR X = 1 TO 1500: NEXT

270  HOME : PRINT  CHR$ (4);"RUN MENU"
```

The CHR$(4) indicates a DOS command.

```
280  SPEED= 255

290  NEW

300  END

310  SPEED= 255

320  INVERSE : HOME

330  FOR X = 1 TO 19

340  PRINT "@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
          @@@@@@@@@@@@"
```

The above line has 40 @ signs, which will fill the screen if you fail the password test in line 200 or 210!

```
350  NEXT X

360  VTAB 23

370  NORMAL : PR# 6
```

Next time I will give you the MENU program mentioned in line 270. It is one of many that enables you to select a file with a single keystroke.

OH MY, OH MY!!!! At least I thought I would give you the program -- I even thought that I would get ahead of the deadline and start on the July column tonight, so I loaded the MENU program (after saving the contents of this article except for the present paragraph) to save it as a text file for use in AppleWriter.

Lo and behold, the program carries a copyright by Bert Kersey from Beagle Brothers. You should all know that Beagle Bros. never, no never (not even harly ever!) protects its programs. But I do not feel that the Cider Press should be active in passing out copyrighted material, except mine (or other C.P. contributors.) So I will have to leave it to you to look up Key-Cat by B. Kersey, found on the Utility City disk.

As it is approaching the summer season, and as I have more than written my 24 columns (TV shows only shoot 22!) I am taking the opportunity to hit the RERUN trail, with a repeat of the column from the Apr-May 1982 issue of the Cider Press.

The quote from Issac Asimov is still applicable, and I imagine that there are enough new members that the column is worth printing again. My offer to share this space with guest contributors still stands, if any of you are anxious to see your deathless prose preserved for posterity!

[Apr.-May 1982]

One of the advantages of writing this column is that I can share opinions as well as instructions. At the latest general meeting there was a brief poll taken of the number of people who were attending for the first time - there was an appreciable number.

It is safe to assume that at least some of them have been asked by their friends, "Why did you buy an APPLE?" While there are probably ALMOST as many reasons as there are APPLES, perhaps one which not many people think of can be shown by analogy to the following quote:

> I imagine that many young scholars have asked themselves rebelliously why on Earth they must learn science when they have no intention of being scientists. Those who feel this way may feel that they need know no more than the minimum that will allow them to just barely get through life.
>
> Why should one know history if one is not going to be a historian, [ed. Aks Santayana!] or geography or languages if one isn't going to travel much?
>
> But surely there is more to life than what one "does". Even if one lives quietly at home and works at some simple, routine job, there must nevertheless be SOME value to understanding the world about us, to understanding events in the light of the past, to having an appreciation of other places and other cultures.
>
> In fact, it is surely FUN to know things. It brightens one's life, sharpens one's wits, reduces one's boredom, broadens one's horizons, makes one more interesting and pleasant to be with. . . .
>
> If, then, you know these other things, do you have to know science, TOO? Is there something special about science?
>
> Actually, there is. . . .
>
> Surely, it will be increasingly important, as the years pass, for people to understand science if they are going to be expected to help make intelligent decisions about how to use science to save the world and not destroy it. That is why it is important to study science even if one is not going to be a scientist.
>
> (Issac Asimov, SCIQUEST, Jan. 1982)

The same argument may be applied to learning about computers. Perhaps one of the things that we, as the "end of the vanguard" of personal computer users, should do is to spread the knowledge of the ways that computers can be used to those who have not yet discovered this mind-expnding capacity.

We also need to know about the "problems" (such as "pirate software") so that we can make an intelligent, and ACTIVE decision about how we will use our computer capabilities. We also need to be careful that, in our enthusiam, we don't snow our friends with acronyms and other jargon.(does CP/M stand for Charlatans Per Minute? Stan Kelly-Bootle says it does in "The Devil's DP Dictionary"!)

Before we can explain to people who are "outside", we need to be able to communicate clearly among ourselves. One of the ways of communicating to others via the computer is to write well-documented programs. There is the "hard-copy" documentation which comes with some commercial programs - everything from a short introduction on the package cover to a fancy padded binder with 100-200 pages and, if you're VERY lucky, a good index.

But we can add documentation to our own programs as we write them. This would be great if you plan to contribute some of your efforts to the SFAC Public Domain Library. Internal documentation consists basically of two different parts of your program; a "dictionary of variables" and "ordinary" REM statements.

"There she goes," you say - "What does REM stand for?" If you have read a tutorial manual, you know it stands officially for REMark (p.63 of the Apple II Plus Tutorial) but it may also stand for REMinder, since I find that I use REM statements to help me follow the logic (it WAS logical, wasn't it?) which I had in mind when I wrote the program.

Some programmers delete REM statements when they are finished with the program because the program executes more rapidly AND takes up less memory space. If you're anything like me, you haven't started writing programs which come even close to using up your 48K so the space problem is immaterial (at the moment) and unless you are writing game programs which require immediate response time in order that the monsters not get you, or unless you have to have your financial information YESTERDAY so that you don't make a wrong investment, the time difference is small enough that it won't be annoying. (You aren't bothered by the difference of milliseconds, are you?)

In order to write a program which is clear to others who are listing it to learn, you MUST plan ahead. Writing a program is like solving a problem or a puzzle, with the logical approach working most of the time and the flash of intuition finishing it off.

The dictionary of variables which I mentioned above is a listing of all the variables which you use in your program and what they stand for. This is a nice extra and it helps you to avoid the use of a variable name for two different things which are not compatible. The lines of this dictionary must be entered as a series of REM statements, not just line numbers with comments on the variables, or you will be attacked by the dread "syntax error" when you attempt to run the program.

Since REM statements are not executed by the program, this dictionary may be placed at the beginning or, if you prefer, at the end. If you choose to place it after the working part of your program you should use an END statement as the last executed line before these REM statements.

As a simple example, let us look at the following program fraagment.

```
10 LET C = 125

20 LET S = 4000

30 LET B = 6*C + 75

40 LET R = B*9
```

While you could easily figure out what B and R will be worth when you run the program, you still don't know what they stand for or what the program will do. Now if you look at the following changes, see what REM statements will do, as well as utilizing the APPLE capability of named (as opposed to designated) variables.

Remember that the APPLE will only recognize as unique the first two letters of any variable name so there must be a judicious use of these names. For examle, you could use APPLE and AEOLIAN, but you could not use APPLE and APPLIED in the same program; you couldn't even use APPLE and APE! Also remember to avoid the use of variable names which contain reserved words (Appendix G, p. 122-123 in the Applesoft Basic Programming Reference Manual)

```
5 REM PROGRAM TO CALCULATE AVERAGE STU-
      DENT BILL AND COLLEGE REVENUE

7 REM

10 REM COST = COST PER UNIT
```

```
20 REM STUDENT = NUMBER OF STUDENTS
       EXPECTED

30 REM BILL = AVERAGE STUDENT BILL: 6 =
       AVERAGE CREDIT LOAD

40 REM REVENUE = REVENUE EXPECTED

50 REM

60 REM USE OF LET STATEMENT IS OPTIONAL
       IN MOST FORMS OF BASIC

70 REM

100 COST = 125

110 STUDENT = 4000

120 BILL = 6*C + 75

130 REVENUE = BILL*STUDENT
```

Notice that you can use a REM statement with nothing after it to cause a "double space" in your listing which sets off the actual program from the documentation. You could accomplish the same thing by typing the following line:

```
70 :
```

This causes the line to be listed in the same manner and also sets off the program from the documentation, but since there are no commands on the line, nothing is executed.

You know that APPLESOFT allows you to have several statements on one line (up to a maximum of 255 characters or a little more than five lines across your screen) so if you wished you could write the above program in the following manner:

```
100 C = 125 : S = 4000 : B = 6*C + 50 :
    R = B*S
```

REM statements may be placed in your program either as stand-alone lines with their own line number(s) or may be placed on a line as one of several statements separated by colons. If you choose to use this method, the program might appear as follows:

```
10 C = 125 : REM C IS COST PER UNIT

20 S = 4000 : REM S IS THE EXPECTED
   ENROLLMENT

30 B = 6*C + 75 : REM B IS THE AVER-
   AGE STUDENT BILL

40 R = B*S : REM R IS THE EXPECTED
   REVENUE
```

However, if you choose this later method, BEWARE!! NOTHING appearing after the three letters R-E-M will be executed. The CPU (Central Processing Unit) will see the REM and go immediately to the next line of the program. So DO NOT combine these two methods!!

What do you think would happen if you tried to execute a program which had been typed in like....

```
10 C = 125 : REM C IS THE COST PER UNIT :
   S = 4000 : REM S IS THE EXPECTED EN-
   ROLLMENT

20 B = 6*C + 75 : REM B IS THE AVERAGE
   STUDENT BILL : R = B*S : REM R IS THE
   EXPECTED REVENUE

30 PRINT B, R
```

The answer is at the end of this column. Don't peek until you've tried to figure it out yourself!

If you use REM statements as stand-alone lines, you may wish to make the comment stand out even more boldly in your listing. You can do this by setting it off with asterisks.

```
10 REM ** THIS IS A TUITION PROGRAM **
```

Other things which are useful in REM statements include the name of the program, the author's name, the date of the latest revision, and the language used in writing the program. If you are still using DOS 3.2, that should certainly be stated.

Try some of these methods of documentation in your programs. They will help you as well as anyone else who wants to see how you did your program and why.

Answer:
The answer to the question above is:

```
      825          0
```

The space between the numbers is caused by the comma in the print statement, line 30. Because Applesoft initializes all variables to zero (and all string variables to null) when you type "RUN", S is automatically zero. Since the program will define C as 125 and then skip to line 20 (NOTHING after a REM statement is executed) that leaves S as zero, and of course B*S will also be zero.

Filemanager/Machine Language Interface

-Bryan Costales

The Final Chapter

(c) B. Costales 1984

ON PROMISES AND BANNERS

I realize that last issue I promised to illustrate FMLI with a Berkeley style, Unix (tm) Cshell. Again (will they never end) I offer apologies. I got so caught up in a "banner" program that I simply did not have the time. Perhaps another day.

If you attended the West Coast Computer Faire, you probably noticed a sign above the Apple Core booth. That banner featured the words "The San Francisco Apple Core" in a large gothic font printed long-ways on fan-fold paper. The program which created that banner (appropriately called "banner") utilizes FMLI.

This is an interesting program as an example of how to use FMLI. It senses whether it is running under DOS or under ProDOS. If under DOS, it coldstarts FMLI and uses FMLI to load the font file from disk. If under ProDOS, it uses the ProDOS Machine Language Interface to load the font.

The approach taken was to assemble FMLI to run at $1000. Since the banner program occupies memory from $803 to $9DF, the two could then be saved together as a single program. The result is a program which can be BRUN under DOS, ANIX, ProDOS, Turbodos, and Diversidos, without modification!

Here, I will give you a mere clipping from the "banner" program. This is in the form of a pseudo assembly language listing.

During your interaction with the program, you are prompted for the name of a font file to load. The monitor routine (GETLN) is used to input the name of the font file at $200 (LINE). GETLN returns the length of the input line in the x-register. This length, needed for Prodos to operate, is stored in $1FF (LINE-$01).

Before the program calls LOADFONT, it first checks to see if it is running under DOS (standard or modified), or under ProDOS.

```
        LDA WHICHDOS
        CMP #PROTOKEN
        BEQ >1
        JSR BFMAIN
```

The variable WHICHDOS is simply the first byt of the ProDOS jump vector at $BF00, and will contain the value $4C if ProDOS is present. If ProDOS is not found, FMLI is cold started with a call to BFMAIN.

Since ProDOS requires a paramater table, and FMLI requires variables, the two are allocated a common area in ram.

```
PARAMTBL HEX 00
FD       HEX 00      ; common to both.
MODEBYTE HEX 00
ERROR    HEX 00
NUMBER   HEX 0400    ; 4 byte header
FONTADR  ADR FONT
         DFS $05     ; fill out paramtbl
```

The routine LOADFONT checks again to see if ProDOS is active, and branches to a seperate section of code if it is.

```
LOADFONT:
         LDA WHICHDOS  ; ProDos?
         CMP #PROTOKEN ; yes, skip.
         BNE *+5
         JMP PROLOAD
;
; the FMLI version of LOADFONT, loads
; a font file at FONT ($2000).
;
; Open the file specified in LINE.
         JSR CALLFMLI
         BYT FOPEN
         ADR LINE
         ADR MODEBYTE
         BCC >1
;
; Carry clear, no error, skip.
; Else carry set, error:
FMLIERR  STA ERROR
; Print the error, returned in a-reg.
^0       JSR CALLFMLI
         BYT FERROR
         ADR ERROR
; Catalog the disk.
         JSR CALLFMLI
         BYT FCAT
; Print a carriage return and abort.
         LDA #CR
         JSR COUT
         JMP ABORT2
;
;Successful open, save the file
; descriptor.
^1       STA FD
;Fetch the file's type.
         JSR CALLFMLI
         BYT FTYPE
         ADR FD
```

```
; strip lock bit
        AND #$7F
; okay if a binary file.
        CMP #$04
        BEQ >2
; any other type, print "bad file type"
; and abort.
        LDA #$06
        JMP FMLIERR
        BNE <0
; Read in the first four bytes of the
; binary file. These contain the ,A$
; and ,L$ information about the file.
; Place these bytes at the location
; pointed to by FONTADR.
^2      JSR CALLFMLI
        BYT FREAD
        ADR FD
        ADR FONTADR
        ADR NUMBER
        BCC *+5
; any error, "error reading file"
        JMP FMLIERR
; successful read. Using the ,L$
; specification now in memory at
; FONT+$02, read in that many bytes
; at the address pointed to by
; FONTADR.
        JSR CALLFMLI
        BYT FREAD
        ADR FD
        ADR FONTADR
        ADR FONT+$02
        BCC *+05
; any error, "error reading file"
        JMP FMLIERR
; Successful load of file, close it.
        JSR CALLFMLI
        BYT FCLOSE2
; Clear carry and return.
        CLC
        RTS
;
; The ProDOS MLI version of LOADFILE.
PROLOAD:
; Fill the paramater table with:
;    param_count = 0A items
;    path_name = LINE-1 (len byte)
        LDA #$0A
        STA PCOUNT
        LDA #LINE-$01
        STA PARAMTBL+$01
        LDA /LINE-$01
        STA PARAMTBL+$02
; Call MLI with command_number and
; address of paramater table.
        JSR CALLPRO
        BYT PGETPOOP ; Get File Info
        ADR PARAMTBL
        BEQ >0
;
; No error skip,
; Else print an error, and abort.
PATHERR  JSR MSG
        BYT "Bad Pathname",CR,0
        JMP ABORT2
;
; Okay, check returned file type.
; $07 is a FNT or font file.
^0      LDA PARAMTBL+$04
        CMP #$07
        BEQ >1
        JSR MSG
        BYT "Not a font type file",CR,0
        JMP ABORT2
;
; Okay, fill paramater list for open.
; PIOBUF is the address of a $1000
; buffer to be used by MLI for the open
; file.
^1      LDA #$03
        STA PCOUNT
        LDA #LINE-$01
        STA PARAMTBL+$01
        LDA /LINE-$01
        STA PARAMTBL+$02
        LDA #PIOBUF
        STA PARAMTBL+$03
        LDA /PIOBUF
        STA PARAMTBL+$04
; call MLI to open the file.
        JSR CALLPRO
        BYT POPEN
        ADR PARAMTBL
; bad open, branch back to error.
        BNE PATHERR
; else fetch and save the
; file descriptor (ref_num).
        LDA PARAMTBL+$05
        STA FD
; Setup paramater table to fetch the
; size of the file (EOF).
        LDA #$02
        STA PCOUNT
        JSR CALLPRO
        BYT PGETEOF
        ADR PARAMTBL
        BEQ >2
; general read error if no success.
READERR  JSR MSG
        BYT "Error reading font",CR,0
        JSR PROCLOSE
        JMP ABORT2
; Transfer length to correct location
; in paramater table, and fill out
; rest of table.
^2      LDA #$04
        STA PCOUNT
        LDA PARAMTBL+$02
        STA PARAMTBL+$04
        LDA PARAMTBL+$03
        STA PARAMTBL+$05
        LDA #FONT
        STA PARAMTBL+$02
        LDA /FONT
        STA PARAMTBL+$03
; Call MLI to read in the file.
        JSR CALLPRO
        BYT PREAD
        ADR PARAMTBL
        BEQ *+5
        JMP READERR
; Error jumps back to read error.
; Successful read, close the file.
PROCLOSE LDA #$01
        STA PCOUNT
        JSR CALLPRO
        BYT PCLOSE
        ADR PARAMTBL
```

```
; and return carry clear if good
; close.
        RTS
```

I hope these examples serve to clarify the approach to using FMLI. As an added bonus, perhaps they may have sparked an interest in learning more about ProDOS's Machine Language Interface. If any interest is expressed via the editor for this rag, I will put together a tutorial series of articles showing the ins and outs of MLI.

ERRATA

The changes to the first listing in part 1 are trival but necessary. If you will recall, I elliminated the use of the break vector in favor of a more usual page-three vector. This necesitated some minor revisions in the source code. The line numbers following, refer to the assembly source listing as it appeared in part 1.

line	action
45	change FCUROFF to FRWSTAT
46	same
49	insert R0 EPZ FCURDAT+$02
	R1 EPZ R0+$02
	R2 EPZ R1+$02
	R3 EPZ R2+$02
56	change BP to PP
57	delete OLDIRQ
79	delete BRKVECT
80	delete DBLVECT
80	add CALLFMLI EQU $3CC
206-215	change installation of break vector to installation of page three vector as:

```
        LDA #$4C
        STA CALLFMLI
        LDA #BFVECTOR
        STA CALLFMLI+$01
        LDA /BFVECTOR
        STA CALLFMLI+$02
```

Since the routine BFVECTOR is no longer arrived at by way of a break interupt, it must be totally rewritten. You will find that it is now much smaller, and easier to understand.

```
;-----------------------------
BFVECTOR:
; Main entry point to fmli.
; Since the number of passed
; arguements will vary, the
; arguement list address is
; kept in PP.
; Saves the registers and
; jumps thru a rts to the
; requested routine.
;-----------------------------
        SEI         ; dissable interupts
        CLD         ; clear decimal mode
        STA ASAVE   ; save the a-reg
        PHP
        PLA         ; save the p-reg
        STA PSAVE
^1      STX XSAVE   ; save x,y.
        STY YSAVE
        PLA
        STA PP
        PLA
        STA PP+$01
        TSX         ; save stack pointer
        STX SSAVE
        JSR INCPP
        LDY #$00
        LDA (PP),Y  ; fetch command token
        BMI >4      ; hi bit on, pos cmd
^3      LDA #EBADCALL
        JMP AERRO   ; "bad system call"
^4      AND #$7F
        CMP #FQUIT-$7F
        BGE <3      ; test command range
        ASL
        TAX
        LDA BFJTABLE+$01,X
        PHA
        LDA BFJTABLE,X
        PHA
        RTS
```

The final bits of change are simply to take all references to BP in the routines BFEXIT, INCBP, DECBP, and PULLADR, and change them to PP. This includes the BP in the subroutine names.

That wasn't so bad, was it? Again, I must apologize for having come up with a new approach in mid-stream. In the long run, however, I am confident these minor changes will make FMLI more powerful.

End of series.

F.M.L.I. is available on disk from the San Francisco Apple Core by request.

An article entitled Disk Customizer written by Bryan Costales appears in the June issue of nibble magazine, Vol. 5/No. 6.

-ed.

MACHINATION

by DON SCELLATO

MACINTOSH "ALERT" BOXES

When you use most computer programs and encounter a problem with a program that is running, a problem that may cause a loss of data, a problem with a mass storage device, or a problem with the computer itself, you usually see a dry message on the screen that announces problem with a one-liner such as "DISK FULL", "BREAK IN LINE 1215", or "OUT OF MEMORY ERROR".

These neat little messages are built in to the program as a warning to the user or are accessible from ROM or from the Disk Operating System. You usually grab the program documentation or a reference manual and try to figure out where to go from the point where everything halted.

If you are using a Macintosh, the process of giving warnings is much different. The Macintosh deals with error messages and dire warnings with a device called an "Alert Box." These Alert Boxes not only warn the user of impending disaster or destruction, they provide him with at least two alternative courses of action and suggest the est alternative.

The discussion which follows delves into how these Alert Boxes are generated and what they mean. The source of the material presented is technical manuals from Apple, magazine articles, and the Macintosh User's Manual. If any reader finds anything that looks technically improper, I would appreciate a call or note explaining the right way of saying what needs to be said.

THE OPERATING SYSTEM AND THE USER INTERFACE

The Macintosh is fashioned in a manner that allows the programmer to use many tools that are available in the computer's permanent memory or ROM. Other tools are not available in ROM, but must be accessed from the routines that have been loaded into the computer's volatile Random Access Memory.

The lowest level tool available to the programmer and to the user is the computer's operating system. This system of machine language routines provides the user and the programmer with interrupt handling, memory management, and Input/Output management.

It also provides for file management, device management, disk interfacing, sound generation, and serial device interfacing. All of these tools are available from the Macintosh's ROM. The tools on the next highest level from the Operating System are called the User Interface Toolbox.

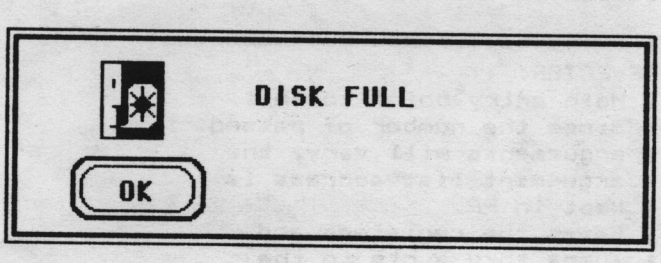
All of the most-frequently used units that make up the User Interface Toolbox are in ROM. They provide the programmer with access to the Macintosh's Standard User Interface. The User Interface provides the link between the program and the user by presenting the user with options, interpreting the user's response to the options presented, and passing the responses back to the program for execution.

The User Interface Toolbox is made up of the Desk Manager, the Scrap Manager, the Dialog Manager, the Control Manager, TextEdit, CoreEdit, the Window Manager, Toolbox Utilities, QuickDraw, the Toolbox Event Manager, the Font Manager, and the Resource Manager.

Each of these tools may be used by a programmer to ensure that the user is treated the same way by most programs that he uses on the Macintosh.

THE DIALOG MANAGER, DIALOG BOXES, AND ALERT BOXES

The Dialog Manager is the utility that provides a Dialog Box to the user when the program requires more information, when something is about to go wrong, or when something has gone wrong with the program or with the computer. A typical dialog box is illutrated below:



The box that is illustrated is the Dialog Box you encounter when you place an unitialized disk in the Macintosh's disk drive. The Dialog Box is a window that contains a pertinent message, a demand for more information, and suggested courses of action.

This Dialog Box informs the user that the disk in the disk drive has not been initialized. It suggests that the disk be initialized by highlighting one of the small boxes within the window. These smaller boxes that are used to issue commands to the computer are called "buttons."

The options that are presented are to initialize the disk or to eject the disk. The user may select an option by placing the mouse pointer on the button he wishes to use and the pressing the mouse button.

Alert boxes are similar to the Dialog Box illustrated and described above. These dialog boxes warn of something potentially going wrong or something gone wrong.

ALERT BOXES

An Alert Box is a special window called a System Window. It surrounds a message that indicates pending or actual trouble. The message which is presented requires an action on the part of the user. Usually only two options will appear in an Alert Box rather than many options as occurs with a regular Dialog Box.

The preferred course of action is usually highlighted by a bold border surrounding the button of the preferred command. The Macintosh's speaker is frequently used as an attention getter when an Alert Box is displayed. When you encounter an Alert Box, don't get rattled.

If you use the recommended alternative, everything will work fine. Typical reasons for Alert Boxes that you as a user might cause to be displayed are attempting to save data to a disk that is full; attempting to save data to a disk drive that doesn't contain a disk; or trying to print data to a printer that hasn't been turned on.

If you see an Alert Box that announces an Address Error, or something as technical sounding as that, you've encountered a problem with the program. The only way out is to reboot the system or return to te Finder. Both of these steps will cause you to lose the changes that you've made. You should note what you did before this happened and communicate your experience to the software manufacturer.

A typical Alert Box is illustrated below:

AN OVERVIEW OF ALERT BOX GENERATION

Rather than get very technical, I will present a brief overview of how the Macintosh generates Alert Boxes. The principles used are similar for other windows and for dialog boxes.

When a program encounters a problem as defined by error trapping procedures written into the program, the Macintosh programmer can cause a predefined Alert Window to be displayed on the screen by using the Dialog Manager.

An Alert Box has a specific Window ID which is passed by the program to the Dialog Manager. Next, the program must send an ID code to the Resource Manager to call out any Text, Graphics, or Controls defined by the programmer by using the Resource Editor to define messages and controls.

These messages, controls, and graphics are stored in an Item List by the Resource Manager. If graphics are required in the Alert Box, Quickdraw is used to draw them. The Item List must also include instructions for resolving the Alert.

All of this activity occurs very rapidly. It causes an Alert Box to be displayed and allows the user to proceed in resolving the alert unless a system or program problem occurs.

CONCLUSION

The Macintosh System User Toolbox Utilities are designed to provide the user with consistent approaches to presenting and executing alternatives when normal activity is being carried out by the computer and when problems are encountered.

These utilities are available to advanced programmers. They provide the user with the convenience of suggested courses of action and familiar screen displays when trouble is encountered.

Machinations

Don Scellato

For those who missed the Macintosh Interest Group Meeting on May 19, it was a very good meeting due to the presentation made by Joe Shelton of Apple Computer. Joe demonstrated MacProject, a PERT chart scheduling program for Macintosh that should be an excellent application for anyone who must draw up complicated schedules and fit complex pieces into a timeline.

Joe also demonstrated MacDraw, a graphics program similar to MacPaint, but more suited to precise engineering or mechanical drawing techniques. Both applications are very exciting and will be treated in more depth in future issues of this column. Harlan Felt demonstrated MacTerminal.

This application will provide a valuable link between Macintosh and other computers either directly or via a modem. All three of these applications will be released in the near future by Apple. Both Joe and Harlan did a wonderful job. They were extremely helpful. Their expalanations of the programs and their demonstration of the programs was extremely valuable to those who stayed for the meeting or who came in for it.

Joe passed on an extremely valuable piece of information that should be shared with everyone who has a Macintosh but missed the meeting. The MacWrite/Macpaint master disk has been updated as have both programs. The system disk now has a four pass disk copy program. The Finder has been enhanced. The "Special Menu" now includes a program for displaying a graphic or text screen as a part of the boot up of a Macintosh disk.

MacPaint has some new fonts and can now print a draft printout of a file. MacWrite can now handle subscripts and superscripts. To get this update, take your original MacWrite/MacPaint disks to your dealer. He will update the disk for you. The next meeting of the Macintosh Interest Group will be held at noon on Saturday, June 23, immediately after our next General Meeting.

If you've got any questions about Macintosh or anything you'd like to share with other Macintosh users, come to the meeting. The remainder of this column will be devoted to using Microsoft Multiplan on the Macintosh to do Moving Average calculations. The column will also describe how to transfer data from one worksheet to another by cutting and pasting.

The Macintosh Version of Multiplan

Microsoft Multiplan is an electronic spreadsheet program that has become very popular on the IBM PC and its clones. This spreadsheet is available in two versions for the Apple II family. It can be used in DOS 3.3 version and a CP/M version on these computers.

Multiplan is popular because it presents its commands on all computers except the Macintosh on a line that runs across the bottom of the screen display. These commands are presented in plain English words rather than as single letter mnemonic abbreviations like those used in VisiCalc, or SuperCalc.

It is also popular on these computers because it adds a third dimension to a spreadsheet by allowing data to be linked to another spreadsheet and loaded into the second sheet automatically. It has a number of other features such as built-in sorting, a clear function, a large variety of output formats, and help screens.

The Macintosh version is much like its predecessors, but it is much more fun to use the mouse rather than a bunch of arrow keys to get the job done.The Macintosh version of Multiplan uses pull-down menus to handle files, edit spreadsheets, format spreadsheets, select options, and select means of calculation.

Multiplan Menus

There are seven pull down menus available for use with Multiplan. These menus are the Apple Menu, the File Menu, the Edit Menu, the Select Menu, the Format Menu, the Options Menu, and the Calculate Menu. The Apple Menu performs the same applications in Multiplan as it does in other Macintosh applications. The File Menu allows the choice of New, Open, Close, Save, Save As..., Page Setup, Print, and Quit for use with spreadsheet files.

The Edit Menu allows the choice of Undo, Cut, Copy, Paste, Clear, Show Clipboard, Paste and Link, Unlink, Paste Name, Paste Function, Absolute Reference, Fill Right, Fill Down, and Sort for use with cells, columns, rows, and sections of a spreadsheet. The Select Menu allows operations to be done with All Cells, Last Cell, Name, and Link.

It also allows the user to Show Last Cell, Define Name, and Delete Name. The

Format Menu allows the choice of General, Dollar, Percent, No Decimal, Decimal, Scientific, and Bar Graph format for numbers. It also allows the number of decimals to be selected. This menu allows text to be Align Left, Align Center, and Align Right.

It allows commas to be used to mark thousands in numbers and allows column width to be reset with the keyboard. (Changing column width with the mouse by moving the marker to the head of a column, pressing the mouse button, and moving the side marker of the column over one way or another is much more fun.)
The Options Menu allows the choice of Freeze Titles, Unfreeze Titles, Set Page Break, Remove Page Break, Show Formulas, Show Values, Protect Document, Unprotect Document, Remove Cell Protection, and Restore Cell Protection. The Calculate Menu allows th choice of Calculate Now, Automatic Calculation, Manual Calculation, Iterate, Don't Iterate, Set Completion Test, and Select Completion Test.

Cells are selected for entry, copying, formatting, protecting, or whaever by placing the plus sign pointer on the cell to be used and clicking the mouse button. This causes the color of the cell to be changed from white to black.

If you wish to indicate a range of cells for any operation, place th pointer on the first cell in the column, row, or block of cells, click the mouse button, and drag the pointer over all of the cells to be used. That's a very short introduction to Multiplan for the Macintosh. Now it's time for a specific application.

Moving Average Calculation

Moving averages are used to smooth curves when stocks and other volatile investments such as commodities are being tracked. A specific period of time is selected as a base for tracking the average activity (high, low, close, or whatever). Data is accumulated over the base period, totalled, and divided by the number of days in the base period to arrive at the average for the period.

It is then tracked daily and the average is computed daily each day over the period ending the day a measurement is taken and divided by the chosen period. Thus a three day moving average would involve accumulating data for three days at the beginning of the measurment taking process, totalling the activity on the third day of measurement, and dividing the total by three.

On the fourth day, the total for that day and the two preceding days is computed and divided by three to arrive at the moving average on that day. This process can be carried out continuously thereafter if a three day moving average process is desired.

Most investors use a 10 day moving average, a thirteen day moving average, a thirty day moving average, or even a two hundred day moving average to smooth the peaks and valleys out of trend analysis. In the interest of simplicity, I've set up a ten day moving average calculation and assumed that a new spreadsheet will be required after the thirty fifth day.

This second spreadsheet will use the data from the last ten days of the prior spreadsheet as starting data.

Starting The Spreadsheet

The attached Figure 1 shows the entire spreadsheet used in this discussion. The headers are entered first by using the Format pull-down Menu to select Center Align after the block of cells ranging from Row 3 Column 1 (R3C1) to Row 4 Column 5 (R4C5) are selected by placing the plus sign cursor on R3C1, clicking the mouse button down, dragging the pointer to R4C5, and releasing the mouse button.

Once these cells have been highlighted, move the pointer to the Format Menu and select Center Align. Next enter the labels used in each cell as shown. The Title at the top was started in R1C2 and enetered continuously. Return was pressed after the word "example" was entered.

The format for Cycle Days was also set at center align although numbers were going to be entered. This was done by highlighting column one and using the Format Menu again. A "1" was then entered in R5C1. A formula was entered in R6C1 to cause 1 to be added to the number in R5C1.

This formula was entered by first entering the equal sign to tell the program to expect a formula. Then the pointer was place on R5C1 and clicked. Then the expression + 1 was inserted after the relative reference shown on the screen. When this method is used, Multiplan issues a "relative reference" value in a formula. This means that the position on the cell being referred to is defined in respect to its position relative to the cell that contains the formula.

This formula would then appear as =R(-1)C+1. Figure two illustrates this formula extended to the thirty fifth day. The R(-1) indicated that the cell referred to is in the row preceding the cell containing the formula. The C indicates that the cell referred to is in the same column as the cell containing the formula.

The formula was then copied from Row 6 to Row 39 in Column 1 by placing the pointer or R6C1, clicking the mouse, dragging the pointer to R39C1, and releasing the mouse button. This causes the cycle days to be numbered without requiring the user to enter each individual number. Next the formulas for deriving the sum of 10 days activity and the average of 10 days activity were entered in R14C5 and R14C4 respectively.

To compute the sum of the range of cells between Row5 and Row 14, the equal sign is first entered in R14C5. Then the word SUM followed by a left parentheses is entered. Then the pointer is moved to R5C3 and the mouse button is clicked. This causes the relative reference R(-9)C(-2) to be ntered in the formula. Then a colon (:) is entered and the pointer is moved to R14C3, the right parentheses is entered, and the Return key is pressed.

This completes the formula for computing the sum of 10 days activity. The formula should then be copied down from R14C5 to R39C5. This is done by placing the pointer on R14C5, clicking the mouse button, and dragging the pointer to R39C5. The mouse button should be released when the pointer reaches that position. Next, the formula for computing the Moving Average must be entered.

Place the pointer on cell R14C4 and click the mouse button. Then enter an equal sign followed by the word SUM and a left parenthese. Then move the pointer to R5C3 and click th mouse button. Then enter a colon. Then move the pointer to R14C3 and click the mouse button again. Then close the parentheses and enter /10 before pressing Return.

Copy this formula from R14C4 to R39C4 the same way the earlier formula was copied. The processes outlined above were all mentioned in the Multiplan manual except for the use of the colon to designate a range for inclusion in a formula. It will set up a simple 10 day moving average calculation. The formulas for columns 4 and 5 are illustrated in illusration 3 below.

Transferring Data from One Worksheet to Another

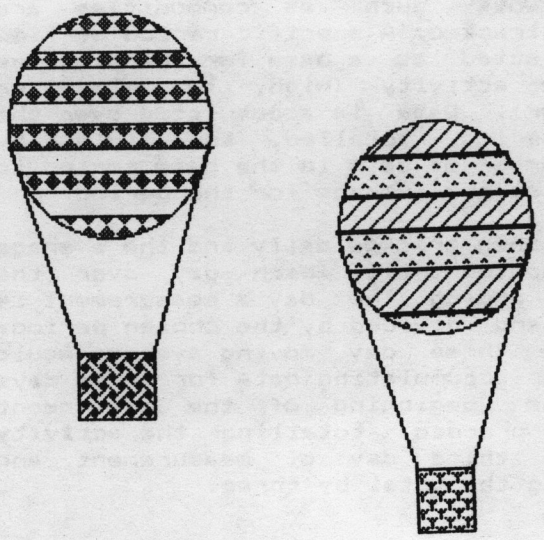
Before entering any data into column 3 of this worksheet, save a blank copy to use as a master for future operations. If you have already entered data, don't worry, you can use the Clear option on the Edit Menu to zero out all of the data not included in formulas on the worksheet being displayed. Enter 35 days worth of data on the worksheet.

Illustration number 4 demonstrates a filled in worksheet. After you've entered the data, save a copy of the worksheet because the next step will remove data from it.Next, place the pointer on R30C3 and press the mouse button. Move the pointer to R39C3 and release the mouse button. Then move the pointer to the Edit Menu and select Cut. See Illustration 5.

Erase the current worksheet by selecting New from the File Menu. Then open the blank master you saved earlier. Then place the cursor on R5C3, press the mouse button, and move the pointer to R14C3. Release the mouse button. Then move the pointer to the Edit Menu and select Paste. See illustration 6.

This will cause the data from the clipboard to be entered into the highlighted column of cells. The result is shown in illustration 7. There are other methods for moving data from one spreadsheet to another. They involve copying rather than cutting data to the clipboard or they involve linking one spreadsheet to another.

If enough of the readers ask for a description, I will be glad to explain the process in a future column. Try what I've described above. It will definitely work.



	1	2	3	4	5	6
1		TEN DAY MOVING AVERAGE EXAMPLE				
2						
3	CYCLE		DAILY	MOVING		
4	DAY	DATE	READING	AVERAGE	SUM	
5	1					
6	2					
7	3					
8	4					
9	5					
10	6					
11	7					
12	8					
13	9					
14	10			0.00	0	
15	11			0.00	0	
16	12			0.00	0	
17	13			0.00	0	
18	14			0.00	0	
19	15			0.00	0	
20	16			0.00	0	
21	17			0.00	0	
22	18			0.00	0	
23	19			0.00	0	
24	20			0.00	0	
25	21			0.00	0	
26	22			0.00	0	
27	23			0.00	0	
28	24			0.00	0	
29	25			0.00	0	
30	26			0.00	0	
31	27			0.00	0	
32	28			0.00	0	
33	29			0.00	0	
34	30			0.00	0	
35	31			0.00	0	
36	32			0.00	0	
37	33			0.00	0	
38	34			0.00	0	
39	35			0.00	0	

fig.1

	1
3	CYCLE
4	DAY
5	1
6	=R[-1]C+1
7	=R[-1]C+1
8	=R[-1]C+1
9	=R[-1]C+1
10	=R[-1]C+1
11	=R[-1]C+1
12	=R[-1]C+1
13	=R[-1]C+1
14	=R[-1]C+1
15	=R[-1]C+1
16	=R[-1]C+1
17	=R[-1]C+1
18	=R[-1]C+1
19	=R[-1]C+1
20	=R[-1]C+1
21	=R[-1]C+1
22	=R[-1]C+1
23	=R[-1]C+1
24	=R[-1]C+1
25	=R[-1]C+1
26	=R[-1]C+1
27	=R[-1]C+1
28	=R[-1]C+1
29	=R[-1]C+1
30	=R[-1]C+1
31	=R[-1]C+1
32	=R[-1]C+1
33	=R[-1]C+1
34	=R[-1]C+1
35	=R[-1]C+1
36	=R[-1]C+1
37	=R[-1]C+1
38	=R[-1]C+1
39	=R[-1]C+1

fig. 2

	4	5
3	MOVING	
4	AVERAGE	SUM
5		
6		
7		
8		
9		
10		
11		
12		
13		
14	=SUM(R[-9]C[-1]:RC[-1])/1C	=SUM(R[-9]C[-2]:RC[-2])
15	=SUM(R[-9]C[-1]:RC[-1])/1C	=SUM(R[-9]C[-2]:RC[-2])
16	=SUM(R[-9]C[-1]:RC[-1])/1C	=SUM(R[-9]C[-2]:RC[-2])
17	=SUM(R[-9]C[-1]:RC[-1])/1C	=SUM(R[-9]C[-2]:RC[-2])
18	=SUM(R[-9]C[-1]:RC[-1])/1C	=SUM(R[-9]C[-2]:RC[-2])
19	=SUM(R[-9]C[-1]:RC[-1])/1C	=SUM(R[-9]C[-2]:RC[-2])
20	=SUM(R[-9]C[-1]:RC[-1])/1C	=SUM(R[-9]C[-2]:RC[-2])
21	=SUM(R[-9]C[-1]:RC[-1])/1C	=SUM(R[-9]C[-2]:RC[-2])
22	=SUM(R[-9]C[-1]:RC[-1])/1C	=SUM(R[-9]C[-2]:RC[-2])
23	=SUM(R[-9]C[-1]:RC[-1])/1C	=SUM(R[-9]C[-2]:RC[-2])
24	=SUM(R[-9]C[-1]:RC[-1])/1C	=SUM(R[-9]C[-2]:RC[-2])
25	=SUM(R[-9]C[-1]:RC[-1])/1C	=SUM(R[-9]C[-2]:RC[-2])
26	=SUM(R[-9]C[-1]:RC[-1])/1C	=SUM(R[-9]C[-2]:RC[-2])
27	=SUM(R[-9]C[-1]:RC[-1])/1C	=SUM(R[-9]C[-2]:RC[-2])
28	=SUM(R[-9]C[-1]:RC[-1])/1C	=SUM(R[-9]C[-2]:RC[-2])
29	=SUM(R[-9]C[-1]:RC[-1])/1C	=SUM(R[-9]C[-2]:RC[-2])
30	=SUM(R[-9]C[-1]:RC[-1])/1C	=SUM(R[-9]C[-2]:RC[-2])
31	=SUM(R[-9]C[-1]:RC[-1])/1C	=SUM(R[-9]C[-2]:RC[-2])
32	=SUM(R[-9]C[-1]:RC[-1])/1C	=SUM(R[-9]C[-2]:RC[-2])
33	=SUM(R[-9]C[-1]:RC[-1])/1C	=SUM(R[-9]C[-2]:RC[-2])
34	=SUM(R[-9]C[-1]:RC[-1])/1C	=SUM(R[-9]C[-2]:RC[-2])
35	=SUM(R[-9]C[-1]:RC[-1])/1C	=SUM(R[-9]C[-2]:RC[-2])
36	=SUM(R[-9]C[-1]:RC[-1])/1C	=SUM(R[-9]C[-2]:RC[-2])
37	=SUM(R[-9]C[-1]:RC[-1])/1C	=SUM(R[-9]C[-2]:RC[-2])
38	=SUM(R[-9]C[-1]:RC[-1])/1C	=SUM(R[-9]C[-2]:RC[-2])
39	=SUM(R[-9]C[-1]:RC[-1])/1C	=SUM(R[-9]C[-2]:RC[-2])

fig. 3

	1	2	3	4	5
1		TEN DAY MOVING AVERAGE EXAMPLE			
2					
3	CYCLE		DAILY	MOVING	
4	DAY	DATE	READING	AVERAGE	SUM
5	1		100.00		
6	2		120.00		
7	3		150.00		
8	4		110.00		
9	5		99.00		
10	6		87.00		
11	7		66.00		
12	8		75.00		
13	9		120.00		
14	10		123.00	105.00	1050
15	11		98.00	104.80	1048
16	12		97.00	102.50	1025
17	13		87.00	96.20	962
18	14		79.00	93.10	931
19	15		93.00	92.50	925
20	16		89.00	92.70	927
21	17		96.00	95.70	957
22	18		90.00	97.20	972
23	19		88.00	94.00	940
24	20		80.00	89.70	897
25	21		86.00	88.50	885
26	22		79.00	86.70	867
27	23		88.00	86.80	868
28	24		86.00	87.50	875
29	25		83.00	86.50	865
30	26		87.00	86.30	863
31	27		90.00	85.70	857
32	28		94.00	86.10	861
33	29		99.00	87.20	872
34	30		100.00	89.20	892
35	31		112.00	91.80	918
36	32		98.00	93.70	937
37	33		94.00	94.30	943
38	34		91.00	94.80	948
39	35		88.00	95.30	953

fig. 4

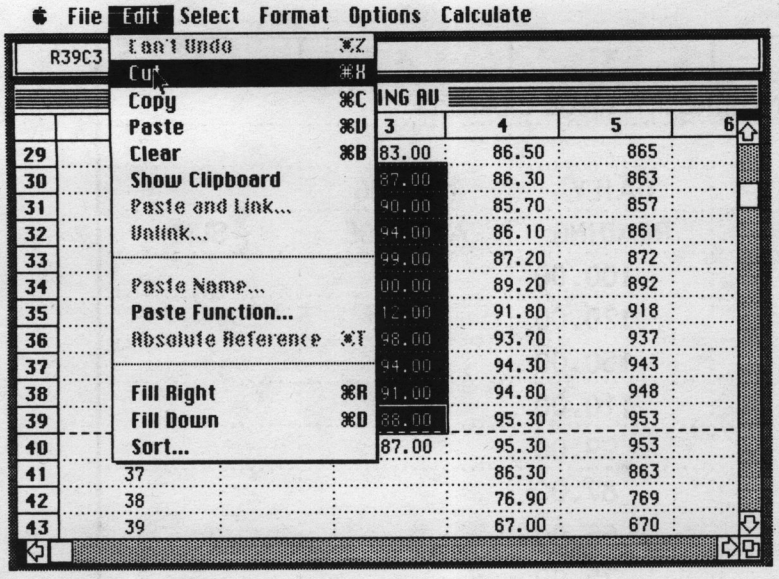


fig. 5

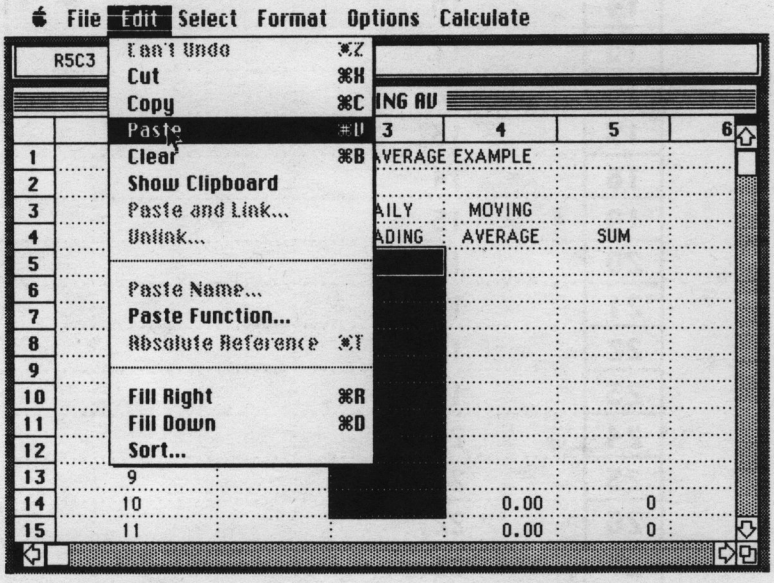


fig. 6

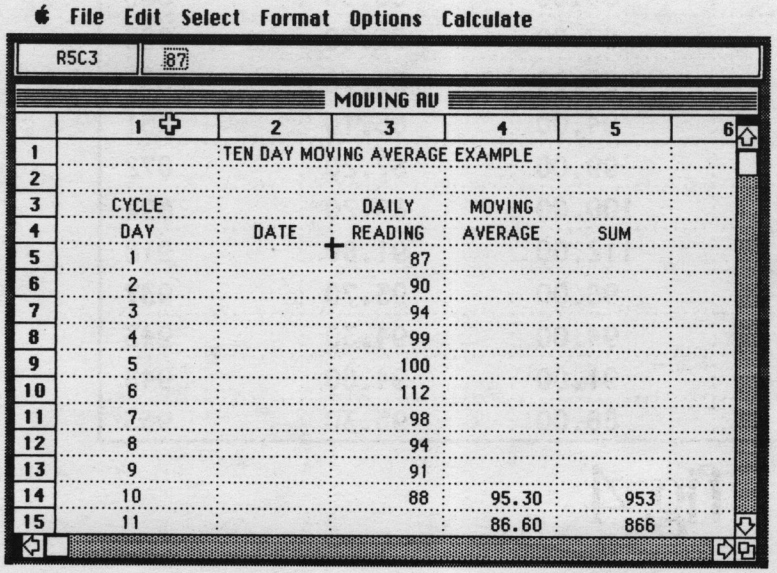


fig. 7

The Brass Apple Award

Don Scellato

Since the San Francisco Apple Core was formed it has been a volunteer organization. The Club started with a bunch of people who each knew something about that very powerful but not very well documented computer called the Apple II. Some of the founders of the Club were programmers who brought in their latest creation and shared it with anyone who was interested. Others were hardware specialists.

They showed their friends and everyone else at the meetings how to hook up disk drives, modems, printers, and whatever. The Club was incorporated and secured a permanent location for a clubhouse. Special interest groups were formed to share experiences with VisiCalc, product reviews, Basic programming, setting up your computer, and CP/M.

The Cider Press changed from a mimeographed newsletter to a more artistic and topical newsletter that was actually printed. The Club collected large amounts of Public Domain software and published it for member use as Disks of the Month. These disks were collected into Libraries that are still maintained at the Clubhouse and are occasionally added to. Meetings became better organized. Guest speakers were invited to address the membership. Manufacturers were invitied to demonstrate their product.

None of these things would have happened without volunteers to make it all happen. These volunteers receive nothing concrete for their time and labor except a sense of helping others to use their Apple computers more fruitfully. Some the volunteers contributed large amounts of time, talent, and energy to getting te Cider Press out, setting up meetings, serving on the Board of Directors, setting up and running a marketing table at meetings, running special interest group meetings, and whatever.

Some of these volunteers burned themselves out or got caught up in politics or personal fallings out with other club members. Most "active volunteers" received a plaque or a certificate at the change of the Board or at the end of their chosen time of participation in club activities. Some of the active volunteers of te past have gone on to contribute significantly to the use and progress of the Apple computer line.

Others have gone on to contribute to the good of the microcomputer industry. Some of the people you see next to you at an Apple Core meeting are or will continue this tradition. Still, within the membership of the Apple Core there are a number of people who hav gone larely unrecognized in te past although they have made significant contributions to the club at large. In the May Board of Diretors meeting, the Board of Directors adopted a motion to find some special way to salute those club members and members of the microcomputer industry who have helped the Apple Core to reach and help a large number of Apple Computer users in the Bay Area.

The salute takes the form of a life size brass apple mounted on a plaque engraved with the individual's name and a plaque that will remain permanently on display in the clubhouse. This plaque will contain the names of the winners of the Brass Apple Award and the year that it was won by the individual. This award may not be awarded by the Board of Directors to any other member of the Board of Directors.

The Board may only give the award to a general member of the club or a member of the microcomputer industry who has given an extraordinary amount of time and energy to helping other club members and to the advancement of the Club. Board members may only be given this award if the Board members other than the selected board member receive a petiton from 25 or more general members.

Two members were selected by the Board of Directors to receive the Brass Apple Award. One of these members was present at the May General Meeting. He was presented with the award at that time. This man, Max J. Nareff, is one of the early members of the club who has maintained his membership through the years that the club has existed.

Max is known as Mister Pascal to those members of the club and those members of the industry who use Pascal. Max has contributed countless programs to the club library and countless articles to the Cider Press. He will spend a few moments or a few hours helping another member with a Pascal programming problem. His advice is always sound and it is freely given.

Max's interests go beyond Pascal to many other aspects of the use of Apple computers. He is a regular attendee of the Macintosh Interest Group meetings. On behalf of the rest of the Board of Directors and of the general membership of the San Francisco Apple Core, I would like to say Thank You and congratulations to Max for being such an active member and such a wonderful person.

I will reserve mention of the second person selected for the Brass Apple Award until the presentation is made to the individual -- hopefully at the June General Meeting. In the meantime, the entire Board of Directors hopes to be making more such presentations to deserving club members.

Systems Technology
MICROCOMPUTER SOFTWARE

2270-I OAK GROVE ROAD
WALNUT CREEK, CA 94598
TEL: (415) 933-1134

Features of INFORMATION MANAGER

Information Manager is supplied on three diskettes; the PROGRAM diskette, the UTILITIES diskette and the SAMPLE DATA diskette. The back side of the UTILITIES diskette contains printer configuration data for 41 models of popular dot matrix printers. The first time you use **Information Manager** you will tell it what printer you have and it will customize itself to support your printer's features.

With the PROGRAM diskette you can: Define your own file structure (up to 30 fields), Add records, Change records, Delete records, List all or some records to screen or printer, Search for records (wildcards characters supported), Sort records, Move files to another disk, Format and print reports to screen, printer or disk, Format data diskettes (with added storage), and more. A screen dump feature is provided when listing or searching for records.

With the UTILITIES diskette you can: Change your file structure (your data files will be altered to the new format automatically), Append files, Copy data diskettes (or other unprotected diskettes), Check the space on any diskette, Obtain helpful information, Print your file structure, Print out your "raw" data, Recover from goofs, and automatically create a database containing all your normal DOS 3.3 diskettes.

With the SAMPLE DATA diskette you can: Breeze through the 25 page tutorial in the **Information Manager** manual, See examples of files, records and report formats, and learn the other program features.

Information Manager supports printers made by the following companies: Anadex, Apple, Axiom, C-Itoh, Centronics, Epson, Integral Data Systems, M.P.I., NEC, Okidata, Star Micronics and Transtar. Letter quality printers are supported in the "generic" mode.

The Manual provides, in addition to a 25 page tutorial, a 60 page Command Reference Guide which explains how to use each command on the Main Menu and the Utility Menu. An introduction is provided with general information. Seven appendicies are provided and include sample program outputs, error codes and recovery procedures, printer information, programmer's information, 80-column card usage (limited), a glossary, and helpful hints. The manual is indexed.

Equipment Required: Four versions of **Information Manager** are currently available: 1) Apple II/II+ with floppy drive, 2) Apple II/II+ with hard drive, 3) Apple //e with floppy drive, 4) Apple //e with hard drive. Apple II models require an Applesoft firmware card or a language card or memory expansion card in slot #0. The Apple II/II+ versions will run on the Apple /// under the emulation mode.

Ordering Information: The price of **Information Manager** is $195. Registered owners of THE INFORMER, THE INFORMER II or THE INFORMER]I[may upgrade to **Information Manager** for $100 by sending their original INFORMER diskette with their order. Mail and phone orders are welcome. VISA, MasterCard, money order, certified check and cashier's check orders will obtain immediate shipment. Personal checks must clear prior to order shipment.

Systems Technology
MICROCOMPUTER SOFTWARE

2270-I OAK GROVE ROAD
WALNUT CREEK, CA 94598
TEL: (415) 933-1134

Features of **ReWrite**

40/70/80 Column Upper/Lower Case Display. ReWrite allows the use of Apple II or Apple II Plus 40 column upper case mode and will capitalize through the use of the ESCape key. It allows use of the Apple //e 40 column upper/lower case with the SHIFT key. It supports the use of a SHIFT key modification on Apple II and Apple II Plus computers. It supports all 80-column cards used with the Apple II and Apple II Plus if the cards support Pascal (almost all do). It supports the Apple //e 80-column card and extended 80-column card. It provides an alternate 70 column high resolution graphic character set.

Editing. ReWrite supports: Automatic or manual search and replace; block move; block delete; block copy; append; screen print; left, right, center and fill justification; headers; footers; margins; boldface; italics; superscript; subscript; underlining; comments; and rapid cursor movement.

Printing. ReWrite supports bold (or shadow) print, italics, underlining, superscript, subscript, variable line spacing, page numbering with different styles, file linking, and allows special printer control codes. As delivered, ReWrite comes with printer control codes for over forty popular models of printers, both letter quality and dot matrix. Printer buffer spooling is supported on the Apple //e or the Apple II and Apple II Plus with a memory expansion card (16k or more).

Spellers. ReWrite supports the use of spelling checkers like Sensible Speller from Sensible Software.

Manual. ReWrite comes with a 35 page manual written in a friendly style.

Mail Merge. ReWrite supports mail merge capability using the data base INFORMATION MANAGER (also from Systems Technology).

Speed. ReWrite is written in 6502 Assembly Language and employs a modified DOS 3.3 operating system to assure fast operation.

Ordering Information. The price of ReWrite is $99. Mail and Phone orders are welcome. VISA, MasterCard, money order, certified check and cashier's checks will obtain immediate shipment. Personal checks must clear prior to order shipment.

ReWrite

Preliminary Review

Allan Woods

Word processing is one of the principal uses to which we put our Apples, and the software to accomplish it ranges from inexpensive and marginally adequate, to full-featured and very expensive.

ReWrite from Systems Technology Software is an attempt to bridge that spectrum with a minimum of trade-offs, many features and a "reasonable" price.

ReWrite will configure to just about any collection of hardware for the Apple. It supports plain vanilla 40 column no *chrome* jobs, Dan Paymar type lower case adapters, modified and/or enhanced keyboards, Shift-key modifications, 80 column cards galore, and the //e.

Specific configuration options within those areas allow you to customize the program to your preferences as well as to your needs.

For those who like the irritating little way the cursor squeezes between letters in AppleWriter][, ReWrite can do it that way too. If you prefer the more reasonable approach of having the cursor directly over the letter to be acted on, you need but configure it so.

If you have no 80 column firmware, there is a 70 column HiRes character generator included.

There are no headaches when it comes to printers either. (This feature is a real beauty!) ReWrite already knows how to work 43 different printers!

No need to fuss over Hexadecimal codes & ASCII values trying to get your printer to do this or that. Just tell ReWrite what printer you have (the name is usually on the hub-caps) and it will do the rest.

Bingo! Underlining, **Bold face**, *Italics*, SuperScript, SubScript. If your printer can do them, so can you.

ReWrite also has nifty provisions for "Print-Time" commands to do a wide range of things from dynamic margin and justification changes to printing messages to the CRT and pausing.

You can chain files to print long documents, or do "MailMerge" and send form letters to your friends and relatives.

There is much more that can be done - headers and footers, page numbering on alternate sides of the page, use printer tab stops, and more.

Two things make this word processor worth a second look (and a more extensive review will be forthcoming): a) It is EASY to use, despite the plethora of features, and b) It costs only $99.00

Systems Technology is in Walnut Creek. If you have questions or need other support, you needn't call New Jersey and be put on hold.

Even if ReWrite isn't the greatest thing since CMOS chips, it's worth checking out.

ReWrite from:
Systems Technology
2270-I Oak Grove Road
Walnut Creek, CA 94598
Tel: (415) 933-1134

LISTEN to "The Voice of Experience"

SynPhonix T.M.

Electronic Speech Articulator for the Apple!

Hear at last, hardware synthesis at software prices. SynPhonix™ produces *unparalleled voice quality* using the ***all new SSI263*** phoneme synthesizer with 64 phonemes, 100's of allophones, plus software controlled pitch, amplitude, rate and filter. The SONIX™ 65© Speech Operating System puts you in control of it all. SynPhonix™ is brought to you by professionals with more than a decade of experience in speech synthesis products. Applications are abundant. SynPhonix™ features include:

- SSI263 Phoneme Synthesizer
- Unlimited vocabulary
- High Quality Phonetic Voice
- Low Data Rate
- Singing/Sound Effects
- Comprehensive User's Manual
- Demo Software Included
- Free Spelling Program
- Transparent to User's Software
- SONIX™ 65©
  - Speech Operating System
  - Speech Editor
  - Preprogrammed Vocabulary
  - Vocabulary File Manager
  - EASY TO USE
- 90 Day Limited Warranty
- Suggested Retail $125.00

Dealer inquiries welcome!

Artic Technologies
"The Voice of Experience"

2234 Star Ct.-2D (313) 852-8344
Auburn Heights, Michigan 48057

Apple is a Trademark of Apple Computer, Inc.

Programming Contest

by
Don Scellato

The San Francisco Apple Core 1984 Programming Contest will start officially on July 1, 1984. It will end on October 31, 1984. All entries must be accompanied by an official entry blank that is available at the Clubhouse if you send a stamped, self-addressed envelope along with a note requesting an entry form.

An entry form will be reproduced in the next Cider Press. Entry Forms will also be available at General Meetings. The programming contest is open only to members in good standing of the San Francisco Apple Core.

All entries must be submitted on disk and must be original programs. In addition to the Entry Form, you should enclose a listing of the program and any documentation you feel is necessary to explain how to use the program and to explain how the program works.

Prizes will be awarded at the November General Meeting and will be announced in the Cider Press that follows the end of the contest. Prizes will be awarded for the following catagories:

1. Educational Programs - Programs that teach a skill to the user. The skill which is taught may consist of any common topic covered by educators or may include the teaching of programming. Programs that are written as management programs for educators will also be included in this category.

2. Management Programs - Programs intended for use in business or home management. This includes spreadsheet applications.

3. Programming Utilities - Programs that aid in the writing of other programs, the management of disk files, or printer utilities.

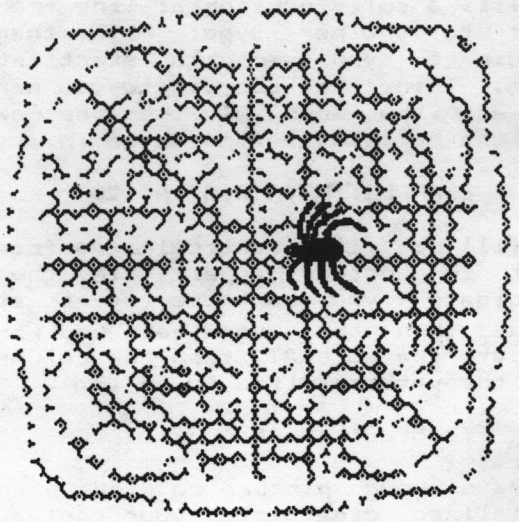
4. Entertainment and Games - Programs written to entertain through game playing, graphics generation, sound generation, or similar use of the computer.

5. Macintosh Applications - Programs of any category written for use on the Macintosh. These programs must be capable of being run with any of the languages used by the Macintosh provided the author is available to demonstrate programs written in Forth, Pascal, or other languages that are not BASIC.

A Grand Prize will be awarded for the best overall program entered in the Contest. The winner of the Grand Prize will not qualify for the first prize of a particular category with the program that wins the Grand Prize.

All prizes submitted become the property of the San Francisco Apple Core for publication in the Apple Core's Public Domain Library and Disks of the Month.

Look for more in the next issue of the Cider Press. By then, we should be able to list a few of the prizes that have been committed by the Industry. In the meantime, get to work on those programs!!!



NUERS

-Kathy Rudovsky

April's Nuer's meeting welcomed guest speaker, Andy Rudovsky, who took those present through an adventurous hand held session of beginning graphics on the Apple IIc. He began the LoRes and HiRes sessions recommending you first draw your picture on graph paper for reference then transfer that information to the computer via low or high resolution graphics modes:

Step by Step LoRes Graphics
by Andy Rudovsky

Load System Master

Type NEW, then press <Return>

Type GR, Press <Return>

If necessary, press the RETURN key enough times until the cursor is seen at the bottom of the screen.

To set the color type: COLOR= <color number>. You have a choice of colors ranging between 0 and 15. Look in your manual for a low resolution color chart.

EXAMPLE: COLOR=5

The Low Resolution Screen is 40 columns down and 40 columns across.

To fill ONE square with a color type PLOT, then the two coordinates for the particular square intended.

EXAMPLE: PLOT 10,10 <RETURN>

To fill a solid horizontal line from one point to another type: HLIN, then the coordinate you want to start at, a comma, then the coordinate you want to end at, the word AT, then the row you want the horizontal line to be in.

EXAMPLE: HLIN 0,39 AT 20

To fill a solid vertical line from one point to another type: VLIN, then the coordinate you want to start at, a comma, then the coordinate you want to end at, the word AT, then the column you want the vetrical line to be in.

EXAMPLE: VLIN 0,39 AT 20

To save your picture to disk, place an initialized disk into your drive and type: BSAVE <name of picture>,A$400,L$400

EXAMPLE: BSAVE ROBOT,A$400,L$400

To bring your saved picture to your screen from the disk type: GR <RETURN>, then BLOAD <PICTURE NAME>

EXAMPLE: GR <RETURN>
BLOAD ROBOT

To clear the picture from the screen and the computer's memory type GR and press <RETURN>.

To cancel LoRes Graphics and get back to a normal screen type TEXT and then press <RETURN>.

Step by Step High Resolution Graphics
by Andy Rudovsky

Load System Master

Type NEW, then press <RETURN>

Type HGR <RETURN>

If necessary press the RETURN key enough times until the cursor is seen at the bottom of the screen.

Type HCOLOR= <color number>. There are 8 colors ranging from 0 to 7. Look in your manual for the HiRes colors available.

To light up a single DOT on your screen type: HPLOT then the two coordinates that correspond to the DOT'S location.

EXAMPLE: HPLOT 0,0 <RETURN>

To draw a line from one place to another type: HPLOT then the starting coordinates, the word TO, then the ending coordinates.

EXAMPLE: HPLOT 0,0 TO 50,50 <RETURN>

To draw a continuous line from one place to another to another to another, ETC....

EXAMPLE: HPLOT 0,0 TO 50,50 TO 75,75 TO 100,100, ETC., <RETURN>

Coordinates should be entered in this fashion:

EXAMPLE: HPLOT 0,0 TO 50,50

1st 0 is vertical coordinate
2nd 0 is horizontal coordinate

1st 50 is vertical coordinate
2nd 50 is horizontal coordinate

Type HGR <RETURN> to clear picture from your screen and the computer's memory.

Type TEXT <RETURN> to return to normal screen.

To save your picture to disk type:
BSAVE <picture name>,A$2000,L$2000 <RETURN>

EXAMPLE: BSAVE ROBOT, A$2000, L$2000

To bring your picture from the disk to your screen type:

HGR <RETURN>
BLOAD <picture name>

EXAMPLE: HGR <RETURN>
BLOAD ROBOT <RETURN>

****

After the bare bones demonstration other ways of getting graphics on the screen were discussed. It was realized by all present that this is just the tip of the iceberg of all there is to know about GRAPHICS.

NUER'S rendezvous on the third Wednesday of each month at 7:30 pm at the SFAC Clubhouse, 1515 Sloat Blvd. #2. Anyone interested or curious is wecome to come. The atmosphere is informal and friendly.

Macintosh Interest Group

Don Scellato

The Macintosh Interest Group of the San Francisco Apple Core held its first meeting on April 14, 1984, at 1 P.M. at Fort Mason, Building C. The meeting was brief.

We discussed the general structure of future meetings and tried to find volunteers to assist with publicizing the meetings, gathering material for the Cider Press, and working with marketing to gather public domain software and seek out bulk buy type bargains.

The next meeting of the group will be held on May 19 at Noon at Fort Mason, Building C. Apple Computer will present two or three new products at this meeting. Meeting structure will consist of a brief question and answer session at the beginning of each meeting and a demonstration for the second half of each meeting.

Interest Group News

Kathy Rodovsky

Two new groups have recently formed and welcome participation of SFAC members, Product Review and Telecommunications.

Angela Allberry is chairing the Product Review Group with the assistance of Robert Kravitz. This group is responsible for contacting manufacturers and software vendors, distributing products to group members - who then use the products and write an article for the *Cider Press.*

The Product Review Group is meeting on Thursday June 7th at 7:30pm at the SFAC clubhouse. Beginning in July, meetings will be held on the 2nd Thursday of each month.

Telecommunications, chaired by George Thornally, is scheduled to meet Tuesday June 12th then again Tuesday August 14th. Please contact George in advance for time and location of the meetings.

East Bay Group meets regularly on the first Tuesday of each month but may be facing a location change so be advised to contact Lenny Leff for meeting confirmation.

If you are interested in learning about C language, Bryan Costales would like to form an interest group and can be contacted at 641-7929.

East Bay Spreadsheets generally meets on the 4th Thursday 7:00 pm at Brueners. Check with Don Scellato for meeting confirmation.

San Francisco Graphics meets the second Monday of each month. The next meeting will be June 11th. Contact Elliott Isenberg at 282-9733 for time and place.

Green Apples an approximately 10 session course in Applesoft Basic, instructed by Keith Thomas, will begin again in the early part of September.

MEMBERS!!! Because most interest groups operate on a drop in basis, please be advised to check with interest group leaders or the SFAC clubhouse office for the most recently updated information regarding interest group meetings.

June

Monday	Tuesday	Wednesday	Thursday
4 App.Basic* 7:30	5 Forth* 7:30 East Bay** 7:30	6 Intro-Forth* 7:30	7 Product Review*
11 Games*	12 Spreadsheets* 7:30 Marin CP/M** Telecomm's**	13 Board Mtg.*	14 Ed. App.*
18 Intro-Pascal* S.F.Graphics**	19 Forth* 7:30	20 Nu'ers* 7:30	21 Intro-Graphics* Lisa**
25 Pascal** 7:30 Intro-Telecomm.**	26 Dot Matrix* Printers Marin CP/M**	27 S.F. CP/M*	28 Intro - Bus. App.*

July

Monday	Tuesday	Wednesday	Thursday
	3 Forth* 7:30 East Bay** 7:30		
9 S.F.Graphics**	10 Spreadsheets* 7:30 Marin CP/M**	11 Board Mtg.*	12 Product Review*
	17 Forth* 7:30	18 Nu'ers* 7:30	19 Lisa**
	24 Dot Matrix* Printers Marin CP/M**	25 S.F. CP/M*	

* At the ClubHouse.

** ElseWhere.

INTEREST GROUP LEADERS

Marin CP/M	Jim Ayers	383-0473(day)
		383-5759(eves)
S.F. CP/M	Ron Dell'Aquila	566-8460
East Bay	Lenny Leff	526-9541
Green Apples	Keith Thomas	837-3164
Dot-Matrix Printers	Michael Fischer	557-1250(day)
		933-4435(eves)
Forth	Robert Dudley Ackerman	626-6295
S.F. Graphics	Elliot Isenberg	282-9733
	Reed Harris	924-7221
Nu'ers	Kathy Rudovsky	756-6480
Product Review	Angela Allberry	654-9351(day)
		391-2715(eves)
	Robert Kravitz	626-5177
Telecommunications	George Thornally	398-7926
		-or- 776-2603
Pascal	Mike Schnapp	386-2238
VisiCalc et al	Don Scellato	672-2051
Lisa	Larry Clopp	333-3707(eves)
Macintosh	Alan Smerican/Don Scellato	566-2342
S.F.A.C. Clubhouse	* * *	566-2342

```
*********************************
*     ------------------        *
*      INVITATION TO JOIN       *
*              a                *
*  C LANGUAGE INTEREST GROUP    *
*     ------------------        *
*                               *
*   Monthly meetings will       *
*  cover such delights as:      *
*                               *
* Dialogs from the UUCP/ARPA    *
*   network's net.lang.c.       *
* Public Domain C programs.     *
* Demystifying C for the        *
*    beginner.                  *
* Obfuscated C code for the     *
*    advanced but foolish.      *
*                               *
* Compilers required:           *
*   Aztec C - 6502 Native       *
*   or BDS C - Z80 CP/M         *
*   Tiny C okay for novice.     *
*                               *
* Contact:                      *
*   Bryan Costales              *
*   author The C Tutorial       *
*          Prentiss-Hall        *
*   641-7929 voice              *
*   641-7947 modem 1200baud     *
*                               *
*********************************
```

VisiCalc Interest Groups

-Don Scellato

The San Francisco Section of the Spreadsheet Interest Group met on Tuesday, May 8 at the Clubhouse at 7:30 P.M. We worked with the Macintosh version of Multiplan. The demonstration convinced one Lotus user to at least consider the Macintosh and this program.

The next two meetings of this group will be Tuesday June 12 and Tuesday July 10. Both of these meetings will be held at 7:30 P.M. at the Clubhouse.

The East Bay Scetion met at 7:00 P.M. at Breuner's in Pleasant Hill on Thursday, May 31. We discussed home checkbook management and the distribution of expenses into categories for tax purposes. The next meeting is tentatively scheduled for Thursday, June 21.

Please check with me before the meeting as Breuner's doesn't always have a room available on the day we want to schedule, so adjustments must be made.

Remember to bring a blank disk to these meetings. We'd be glad to share any spreadsheet applications we have and would like copies of those you are using. Bring along any applications that may be giving you problems. We'd be glad to help you find a solution to those problems.

The Spreadsheet Interest group met in the East Bay at Breuner's in Pleasant Hill on Thursday, May 26, at 7 P.M. The Apple //c was presented to the group and two integrated software programs were examined.

These programs were JANE, a mouse-oriented program, and AppleWorks, an arrow key-oriented program. The next East Bay meeting will be held at Breuner's in Pleasant Hill off of Buskirk at 7 P.M. on Thursday, May 31.

The June meeting is tentatively scheduled for Thursday, June 28. This date is indefinite as Breuner's only allows the room to be reserved about thirty days prior to its use. Check with me prior to the June meeting if you are interested in attending.

The San Francisco branch of the Spreadsheet Interest Group met at 7:30 P.M. on Tuesday, May 8, at the Clubhouse. Appleworks and The Dow Jones Spreadsheet Link were discussed. The next meeting of this group will be held on Tuesday, June 12 at 7:30 P.M. at the Clubhouse at 1515 Sloat Boulevard.

We encourage new and experienced spreadsheet users to attend these meetings. Bring any applications you are currently using and share them with the rest of the people who attend. If you are having problems making your spreadsheet program do certain things, come to a meeting and we'll help you solve the problems.

Poke of the Month

Kathy Rudovsky

Thanks to Beagle Bros. Peeks, Pokes and Pointers chart try POKE 44505,234 or POKE 44506,234 then type CATALOG. This POKE will expose file names in your catalog listing that have been deleted.

Bus Service and Directions to Fort Mason Center

Fort Mason can be reached by the following Muni lines.

#22 — Fillmore (Fillmore and Laguna)

#28 — 19th Ave. (terminates inside west gate of FMC)

#30 — Stockton (Laguna and Chestnut)

#42 — Downtown Loop (VanNess and Bay)

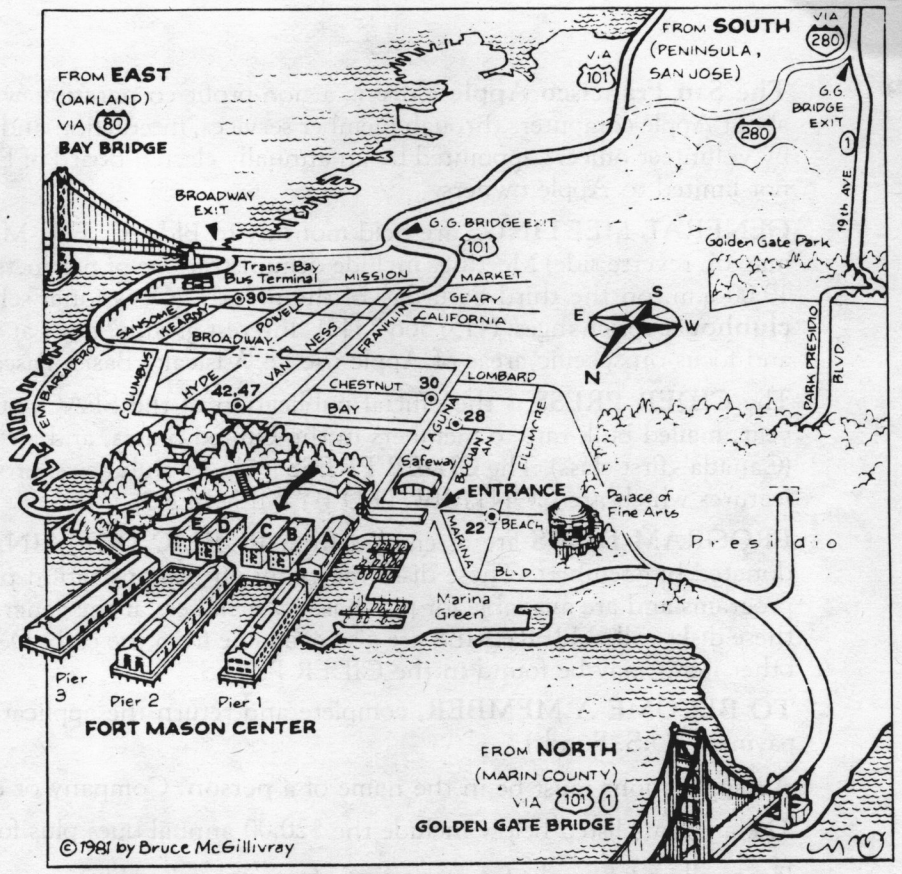
#47 — VanNess (VanNess and Bay)

These are the bus stops closest to FMC. For further information call 673-MUNI.

Golden Gate Transit buses intersect #28-19th Ave. buses at the S.F. G.G. Bridge toll plaza. Golden Gate Transit — 332-6600.

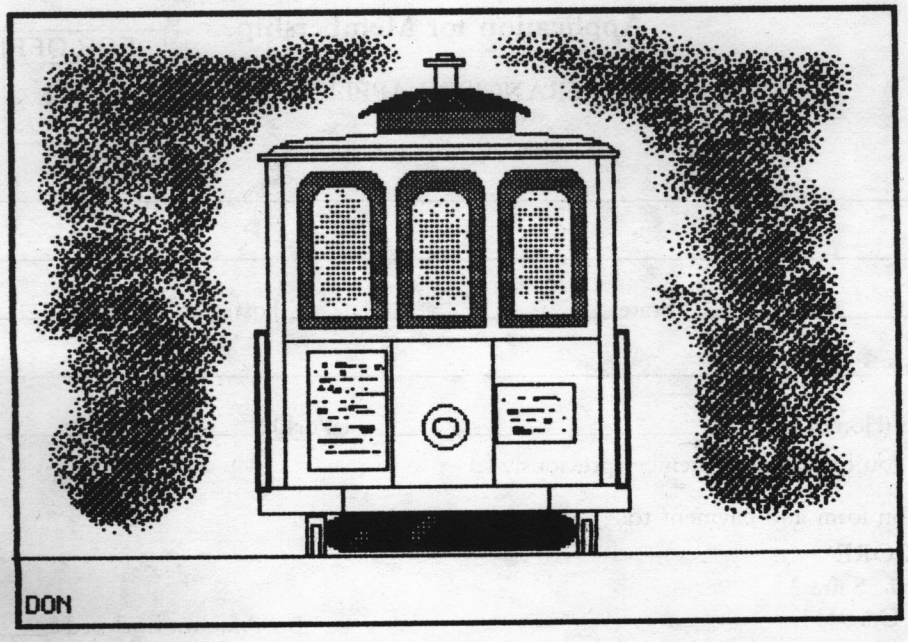
DIRECTIONS: **EAST**—Bay Bridge to Broadway Exit—Broadway to Laguna, Rt. turn to Marina Blvd. & Buchanan St.—Rt. turn. **Peninsula**—U.S. 101 North to Franklin St. Exit. Franklin to Bay, lft. turn to Laguna, Rt. turn to Marina Blvd. Rt. turn at Buchanan.

Pedestrian entrance at Laguna. Auto entrance at Buchanan.



⊙ - BUS STOP **22**/ Fillmore **28**/ 19th Ave. **30**/ Stockton **42**/ Downtown Loop **47**/ Potrero

FOR MEETING INFORMATION CALL (415) 566-2342



San Francisco Apple Core, Inc.

1515 Sloat Blvd., Suite 2, San Francisco, CA 94132

(415) 566-2342

The San Francisco Apple Core is a non-profit corporation whose purpose is to share information about Apple computers through member services, interaction and the club newsletter. The SFAC is run by volunteer officers appointed by an annually elected Board of Directors. Membership in the SFAC is not limited to Apple owners.

GENERAL MEETINGS are held monthly in Bldg. C, Fort Mason Center, San Francisco, CA (see map on reverse side) Meetings include demonstrations of products and techniques and are scheduled at 10:00 a.m. on the third Saturday of the month. (Occasional scheduling conflicts may occur - phone **clubhouse** for changes. (415) 566-2342) "Interest groups" meet at various locations in the S.F. Bay Area and focus on specific areas of Apple use i.e. Visicalc, Basic, Pascal, etc.

The **CIDER PRESS** is the official publication of the SFAC, published approximately 10-12 times a year, mailed bulk rate to members in the United States, and air mail to members in foreign countries (Canada - first class). The **CIDER PRESS** includes club news, program listings, tips, reviews and special features which are CONTRIBUTED BY MEMBERS.

PROGRAM DISKS are assembled from **PUBLIC DOMAIN, NON-COPYRIGHTED** programs donated by members. These disks may contain demonstration programs, utilities, games, educational programs and are available for purchase by members at the general meetings or by mail. Collections of these disks called LIBRARIES are also available from the club. Details regarding purchases of these and other items may be found in the CIDER PRESS.

TO BECOME A MEMBER, complete and return the application form below with the appropriate payment (U.S. Funds).

All applications must be in the name of a **person**. Company or organization names are not accepted.

The amounts listed below include the $20.00 annual dues plus foreign mailing costs where applicable.

Please allow 6-8 weeks for processing of your membership.

USA	$20.00	South America, Europe, North Africa	$41.00
Canada	$30.00	Eastern Europe, Asia, Pacific Isles	$47.00
Mexico, Carribean, Central America	$30.00	Middle East, Australia, South Africa	$47.00

Detach along dotted line

Application for Membership

SAN FRANCISCO APPLE CORE

FOR OFFICE USE ONLY		

Name _____

Address _____

City _____ State _____ Postal Code _____

USA ☐ Other _____

Phone Number (Home) _____ (Work) _____

Check here if you have been a member previously ☐

Send application form and payment to:

S.F. APPLE CORE
1515 Sloat Blvd., Suite 2
San Francisco, CA 94132

Amount Enclosed $_____

[Rev. 1/84]

SF APPLE CORE
1515 Sloat Blvd., Suite 2
San Francisco, CA. 94132

TIME VALUE

Bulk Rate
U.S. Postage
PAID
Permit #85
El Cerrito, CA 94530

TO:

3121AUG84
DOUGLAS STORKOVICH
921 PINTO CT
WALNUT CREEK CA 94596